25 maggio 1988 - Anno VII - N. 53
Sped. Abb. Post. Gr. III/70
GR - Distr. MePe

Amiga e C64 in "C", un solo linguaggio

Videogame da incubo

DYLAN DOG — LE NOTTI DELLA LUNA PIENA

Con Licia la tua rivista in TV

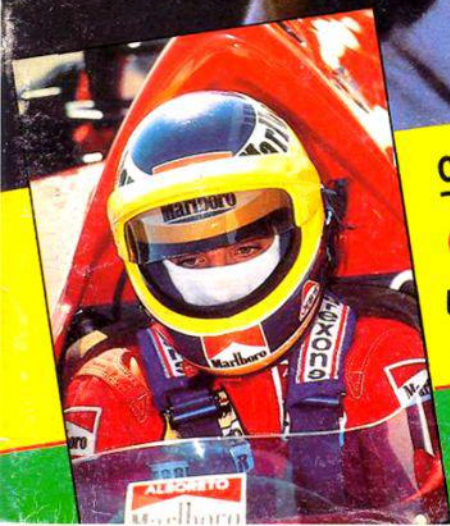
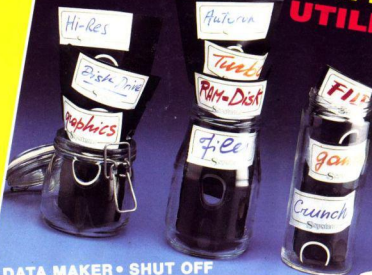
Esclusivo! Scriptor, un word processor per C16

BIT PARADE
20 nuovi giochi da sballo

CX per 64 **Ferrari Testarossa contro Mitsubishi Pajero**

Un programma per misurare il coefficiente aereo dinamico della tua auto

CAMPUS
32 PAGINE GIOCANDO PER IMPARARE

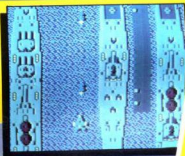
Sommario

RUBRICHE

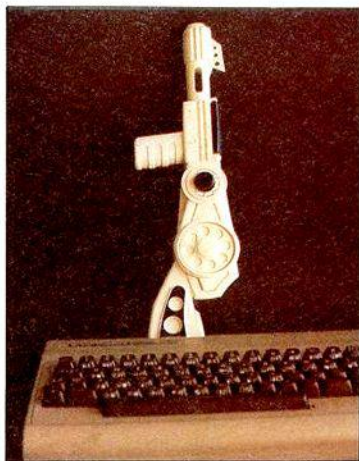
4 EDITORIALE

5 LA VOSTRA POSTA

105 I SUPERGIOCHI DEL MESE

109 GUIDA ALL'ACQUISTO

112 I COMMODORE POINT

PAG.	REMarks	C64	C128	C16	Amiga	Gener.
	Anteprima					
17	Un dischetto da incubo	•				
	L'Utile					
19	Scriptor: un word processor per C16			•		
100	Dallo schermo alla stampante		•			
	Applicazioni					
27	Il CX questo sconosciuto	•	•	•		
	Linguaggio C					
33	Come funziona il "C"				•	
	Hardware					
37	Una memoria di ferro	•				
81	L'epopea del Commodore 64	•				
	Attualità					
77	Se potessi avere 1000 lire al mese...	•	•	•		
	Enciclopedia di routine					
85	Guarda come ti sposto il video	•				
	Rassegna di effetti grafici					
92	Alla scoperta degli istogrammi con i "piccoli" Commodore			•		
	CAMPUS: inserto speciale per piccoli Commodore					
I/43	Il mezzo grafico	•	•	•		
IV/46	Al di là delle sbarre del video	•				
XIII/55	Movie Maker, secondo tempo	•				
XIX/61	Tutti i numeri del vostro C 64	•				
XXXIII/65	Invito alla Kermesse del Kernel di Amiga				•	
XXIX/71	Se la memoria non vacilla	•				

Direttore: Alessandro de Simone -
Redazione/collaboratori: Paolo Agostini, Davide Ardizzone, Claudio Baiocchi, Luigi Callegari, Michele Maggi Sergio Camici, Umberto Colapicchioni, Maurizio Dell'Abate, Valerio Ferri, Roberto Ferro, Cristina Magnaghi, Giancarlo Mariani, Roberto Marigo, Clizio Merli, Marco Mietta, Marco Miotti, Oscar Moccia, Roberto Morassi, Guido Pagani, Antonio Pastorelli, Sonja Scharrer, Fabio Sorgato, Valentino Spataro, Danilo Toma
Segretaria di redazione: Maura Ceccaroli **Grafica:** Arturo Ciaglia
Direzione, redazione, pubblicità: v.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
Pubblicità: Milano: Leandro Nencioni (direttore vendite), Guido Agosti, Giorgio Ruffoni, Viale Famagosta, 75 - 20142 Milano - Tel. 02/8467348
• Emilia Romagna: Spazio E - P.zza Roosevelt, 4 - 40123 Bologna - Tel. 051/236979
• Toscana, Marche, Umbria: Mercurio srl - via Rodari, 9 - San Giovanni Valdarno (Ar) - Tel. 055/947444
• Lazio, Campania: Spazio Nuovo - via P. Foscari, 70 - 00139 Roma - Tel. 06/8109679
Segreteria: Tiziana Sodano - **Abbonamenti:** Liliana Spina
Tariffe: prezzo per copia L. 4.500. Abbonamento annuo (11 fascicoli) L. 45.000. Estero: il doppio. Abbonamento cumulativo alle riviste Computer e Commodore Computer Club L. 85.000.
I versamenti vanno indirizzati a: Systems Editoriale Srl mediante assegno bancario o utilizzando il c/c postale n. 37952207
Composizione: Systems Editoriale Srl - **Fotolito:** Systems Editoriale Srl
Stampa: Systems Editoriale
Registrazioni: Tribunale di Milano n. 370 del 2/10/82 - Direttore Responsabile: Michele Di Pisa
Sped. in abb. post. gr. III - Pubblicità inferiore al 70% - **Distrib.:** MePe - via G. Carcano, 32 - Milano
Periodici Systems: Banca Oggi - Commodore Club (disco) - Commodore Computer Club - Commodore Computer Club (disco produzione tedesca) - Computer - Computer disco - Electronic Mass Media Age - Energy Manager - Hospital Management - MondoRicambi - Nursing '90 - PC Programm (disco) - Personal Computer - Security - Software Club (cassetta ed. italiana) - VR Videoregistrare

COME TI DIROTTO L'AEREO

E' mai possibile che, in piena era tecnologica, possano salire, a bordo di un aereo, pistole, mitra e bombe senza che nessuno se ne accorga? La risposta è negativa; a meno che...

I moderni sistemi di allarme, come è noto, non lasciano possibilità di equivoci, e se sbagliano, semmai, è per eccesso di prudenza. Un sensore di fuga di gas, se realizzato con cura, lancia l'allarme perfino nelle nostre cucine, nel caso vi siano perdite, oppure se il latte, sbollendo, ha spento la fiamma.

Al "check in" degli aeroporti, non c' valigia, zaino e borsetta il cui contenuto possa sfuggire all'impietoso occhio dei sensori a raggi, sistemati agli ingressi.

Il problema, come al solito, non risiede nei sofisticati strumenti cui si affida il delicato compito di sorvegliare, ma nell'uomo addetto a farli funzionare.

"Chi sorveglia il sorvegliante?". E' questo, infatti, l'unico, spinoso problema che non potrà mai esser risolto in nessuna epoca pur disponendo dei prodigiosi strumenti che verranno realizzati.

E' sufficiente un solo buco nella maglia (ripeto: uno solo) perchè tutto il sistema di sicurezza fallisca miseramente.

E' sufficiente che un poliziotto, un facchino, un pilota, un'hostess oppure uno qualunque della miriade di persone che operano di solito all'interno di una struttura aeroportuale, apra un cancello laterale, un passaggio poco sorvegliato, un cunicolo scoperto chissà come: ed ecco che passa la pistola, il mitra, la bomba. Il sistema elettronico posto al check in, nel frattempo, è indaffarato a insospettirsi per macchine fotografiche, pezzi di argenteria e altri souvenir metallici ingenuamente nascosti in un doppio fondo.

In una struttura in cui operano decine di persone, e che viene utilizzata quotidianamente da migliaia di individui, è praticamente impossibile imporre una rigidità di controlli a tappeto, tale da impedire sorprese.

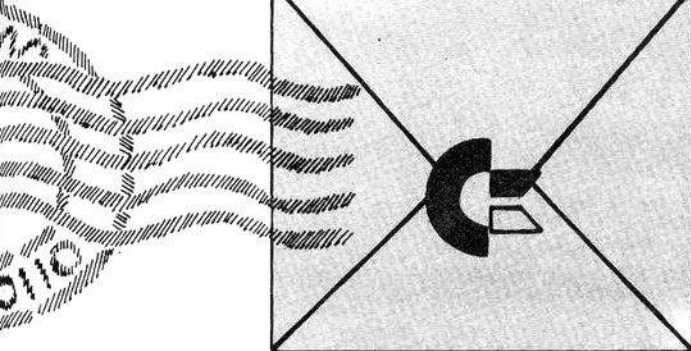
E poi, anche se riuscissero a realizzare una protezione di questo tipo in un aeroporto, gli "interessati" potranno sempre passare attraverso qualche maglia lasciata aperta in un altro aeroporto, uno scalo dopo, o uno scalo prima.

Stesso discorso vale per i traffici di armi ed altri traffici con elevato "valore" commerciale. E' assolutamente inutile illudersi di controllarli: gli interessi in gioco sono troppo alti. Approvare leggi che consentano di fabbricare ed esportare armi, ma solo a paesi il cui elenco è stabilito da apposite commissioni, impedisce soltanto di abbreviare il percorso. Trucchi, più o meno legali, per far giungere il carico di armi nel paese "vietato" ve ne sono adesso e, ciò che più è grave, ve ne saranno ancora, e tutti lo sanno.

Un mezzo più efficace sarebbe la proibizione globale della fabbricazione di mine, bombe, aerei, carri, in modo che qualunque carico di armi possa esser subito considerato illegale. Ma questa soluzione, si sa, è contraria a troppi interessi commerciali (e non).

La tecnologia più o meno raffinata, pertanto, non riuscirà mai a correggere le storture mentali dell'uomo.

Non prendiamocela, quindi, con un computer che, poveretto, non c'entra per nulla nelle beghe umane di basso livello.

Alessandro de Simone

la vostra posta

Emulatore C/64 per Amiga

☐ L'Amiga è totalmente compatibile con il C/64 utilizzando l'apposito emulatore?
(Salvatore Caputo - Sciacca)
(Gabriele Tramonte - Carbonia)

• Un computer molto potente, di solito, dovrebbe essere in grado di riprodurre le potenzialità di elaboratori "minori".

Il microprocessore che impiegano, infatti, consente di imitare tutte le funzioni di micro più modesti, ad una velocità, magari, maggiore.

Tuttavia il programma di gestione di una simile utility deve essere tale da tener presente la più piccola differenza che, a livello di linguaggio macchina, rischia di inchiodare il computer se malamente impostata.

Di emulatori C/64 per Amiga sono in commercio diverse versioni, alcune più datate, altre più recenti. Tutte, comunque, richiedono indistintamente il drive 1541 per funzionare (altrimenti come si potrebbero caricare i programmi del C/64?) e tutte, allo stesso modo, presentano malfunzionamenti in alcuni programmi un po' particolari, come giochi in l.m. oppure programmi protetti in modo inconsueto.

Ciò non significa che un 68000 non sia in grado di rendere efficiente, su Amiga, una struttura relativamente semplice come quella impostata sul C/64 da un 6510; significa, invece, che i progettisti dell'emulatore possono non aver considerato alcuni casi particolari, che tuttavia sono la base di package sofisticati.

In conclusione possiamo affermare che il Basic del C/64 viene mantenuto efficace e funzionante al 99.99% Tale percentuale diminuisce a mano a mano che si desidera utilizzare Poke un po' al di fuori del normale, Sys ancora più strane e programmi decisamente fuori dell'ordinario.

Il Top di memoria

☐ A che serve e come si usa il programma "Fissa Top di Memoria" che suggerite sempre di utilizzare prima di attivare routine in linguaggio macchina (tra cui le routine dell'Enciclopedia?)
(Vincenzo Valle - Campobasso)
(Luciano Riccardi - Roma)

• Quando si accende il C/64 la memoria disponibile è di 38 KRam circa. A volte è necessario effettuare una partizione di tale memoria, nel senso di lasciarne una parte al Basic e la rimanente per "ospitare" una pagina grafica, i dati degli sprite oppure routine in linguaggio macchina.

Si potrebbe pensare, erroneamente, che se un programma Basic non occupa tutta la memoria a disposizione, ne dovrebbe conseguire, automaticamente, che la parte rimanente è disponibile per qualsiasi altro uso.

Ciò, purtroppo, non si verifica perchè un programma Basic non si limita ad occupare l'area Ram per allocare... se stesso ma, non dimentichiamolo, anche una zona, impossibile da individuare a priori (se non con molta difficoltà) in cui memorizzare i valori delle variabili numeriche e stringa.

Mentre, però, le prime vengono sistemate automaticamente in coda al programma, a mano a mano che vengono definite, le variabili stringa vengono invece memorizzate a partire dal fondo della memoria (indirizzi più alti) verso il basso (limite dell'area occupata dal programma Basic e delle variabili numeriche).

Un eventuale programma l.m. scritto in fondo, quindi, verrebbe inesorabilmente cancellato dalla gestione delle stringhe, con effetti facilmente immaginabili.

Per evitare questo è sufficiente servirsi di aree che il Basic non "tocca" mai (come quella posizionata a partire da 49152) oppure si può ricorrere al trucco dell'alterazione del Top di memoria.

Il computer, infatti, "crede" che la memoria Ram disponibile per il Basic termini nella locazione indicata (nella solita forma byte basso / alto) dai byte 55 e 56 e si "arrangia" di conseguenza, in modo totalmente automatico. All'accensione del C/64 i due byte contengono, rispettivamente, i valori 0 e 160 che, appunto, forniscono 40960 (0 + 160*256); tale valore, decrementa-

Aiutateci a servirvi meglio

• Spesso alcuni lettori, che dichiarano di possedere numeri arretrati della nostra rivista, porgono quesiti le cui risposte sono già state esplicitamente pubblicate (in occasione di risposte ad analoghe domande) oppure sono contenute in articoli presenti nei fascicoli in loro possesso.

Per evitare di ripetere argomenti già trattati, pertanto, ricordate di indicare sempre, nelle lettere che ci inviate, i numeri dei fascicoli in vostro possesso: potremmo infatti indicarvi, se esistono, gli articoli che, in un modo o in un altro, possono chiarire gli argomenti richiesti.

Interfacce per Mps-1200

□ Il mio negoziante mi aveva assicurato che con la Mps-1200 avrei potuto usare il mio C/128, e in seguito (con opportuna interfaccia) anche l'Amiga; purtroppo, dopo oltre due mesi di attesa, non posso ancora stampare nulla con l'Amiga perchè l'interfaccia non risulta ancora disponibile.
(Aniello Rania - Aversa)

• Il nostro lettore non indica il negoziante presso il quale è stato effettuato l'acquisto. E' molto probabile, tuttavia, che non sia un punto di vendita molto specializzato nel campo dei computer.

Da parecchio tempo la Commodore suggerisce ai suoi clienti di rivolgersi ai "Commodore Point", veri e propri centri commerciali Commodore, che godono di un trattamento di favore da parte della casa madre.

Molto spesso, infatti, negozianti "generici" incontrano comprensibili difficoltà nel procurarsi accessori poco diffusi; tale inconveniente, al contrario, non si verifica presso i Commodore Point che, in breve tempo, possono procurarsi tutto ciò che risulta presente nel listino ufficiale della Commodore Italiana (oltre a numerosi altri prodotti).

Ti consiglio, pertanto, di contattare il Commodore Point più vicino alla tua città o di telefonare alla Commodore Italiana (tel: 02/61.83.21).

Hai provato, inoltre, a contattare i nostri inserzionisti, la cui pubblicità appare periodicamente sulla nostra rivista? Ti posso assicurare che sono fornitissimi ed evadono ordini anche per corrispondenza.

to di un'unità (cioè: 40959) rappresenta l'ultima locazione gestibile dal Basic.

Prova, infatti, a seguire i passi che ti suggerisco:

• accendi il computer
• batti:
Poke 40959,0
• chiedi conferma con:
Print Peek(40959)
• definisci la variabile A$:
A$="A"
• chiedi nuovamente:
Print Peek(40959)

...avrai in risposta un sorprendente: 65 che è il codice Ascii della variabile A$. Provando, subito dopo, a digitare:

Poke 40959,66: Print A$

...avrai in risposta "B" e non più "A" proprio perchè il codice di A$ è, ormai, 66 e non più 65.

Questa esperienza dimostra che il Basic utilizza, per le stringhe, la parte alta della memoria il cui limite, lo ripeto, ritiene sia fornito dal calcolo delle locazioni 55 e 56.

Alterando opportunamente il loro contenuto, quindi, è possibile ingannare il calcolatore facendogli credere di poter disporre di una quantità di memoria più modesta.

Un programma che alteri le locazioni 55 e 56, quindi, (come, appunto, "Fissa Top di Memoria") sembra che non sortisca alcun effetto mentre, te lo assicuro, provvede a rendere "invulnerabili" le routine l.m. che vorrai trascrivere nella memoria così liberata.

E poi non è affatto vero che non sortisca alcun effetto: prova a chiedere la quantità di memoria disponibile (Print Fre(0)) prima e dopo aver dato Run al programma!

Amiga 1000

□ Dopo la presentazione dei modelli A-500 ed A-2000 che cosa accadrà al mio A-1000, che mi affrettai ad acquistare appena disponibile in Italia?
(Massimiliano Agresti - Grumolo)

• Purtroppo non sarà facile rendere il vecchio A-1000 totalmente identico al nuovo A-2000, nonostante conservi la compatibilità (a patto di servirsi del Kickstart V 1.2).

Il fatto è che, nel prossimo futuro, eventuali nuove versioni di Sistemi Operativi per Amiga prenderanno, come riferimento, le potenzialità dei modelli "correnti" che dovrebbero rappresentare la versione definitiva (per ciò che riguarda la compatibilità in "salita").

A-1000, tuttavia, dovrebbe continuare a dare le soddisfazioni che finora hai ottenuto, a parte la seccatura di esser costretti a caricare due dischi prima di iniziare a lavorare.

Mouse vantaggioso

□ Quali vantaggi posso ottenere collegando il mouse al mio computer?
(Andrea Trento - Parabiago)

• Il mouse è uno strumento semplice e rivoluzionario che permette un rapidissimo "dialogo" tra utente e computer.

I nuovi computer (Amiga, OS/2), ed i loro sistemi operativi (S.O.), prevedono il mouse e lo considerano alla stessa stregua della tastiera. Naturalmente i progettisti dei S.O. hanno inserito, tra le varie routine, anche quelle che si "accorgono" del movimento del mouse, della pressione dei suoi tasti, e si comportano di conseguenza caricando programmi, visualizzando finestre, eseguendo comandi e così via.

All'epoca della progettazione del C/64 e del C/128 il mouse non esisteva ancora (o meglio: esisteva, ma era molto costoso) ed il sistema operativo di questi computer, quindi, era (ed è tuttora) in grado di ricono-

scere, come input, solo la tastiera.

Le software house che diffusero, in seguito, la gran quantità di programmi che ben conosci, sfruttarono abilmente le altre porte disponibili, prime tra tutte quelle relative ai joystick. Altri accessori furono costruiti grazie alle potenzialità offerte dalle porte joy: penne luminose e varie tavolette grafiche.

Si trattava (e si tratta), comunque, di accessori che sono "riconosciuti" dal computer solo se è attivato un programma che ne preveda il collegamento: da soli non servono assolutamente a nulla.

Mentre, quindi, con l'Amiga e con i computer della serie OS/2 il mouse è sempre "attivo", con il C/64 ed il C/128 è necessario utilizzare programmi specifici.

Al momento attuale l'unico package che ne prevede esplicitamente l'applicazione è il Geos che è, in realtà, un vero e proprio sistema operativo che si sostituisce interamente a quello standard della macchina. Molti giochi che richiedono il joystick, tuttavia, funzionano anche se viene collegato, invece del joy, il mouse, senza apportare modifiche di alcun genere; in alcuni casi il mouse si dimostra migliore del joy, in altri, invece, presenta difficotà di gestione che ne sconsigliano la sostituzione.

In ogni caso si tenga presente che il mouse non è altro che un joy a "ripetizione" e viene gestito anche da Basic, con le stesse modalità con cui viene gestito il movimento dell'asticella del joystick.

Antiformat

☐ Il programma Antiformat (C.C.C. n.49), che consente di recuperare i file di un dischetto formattato accidentalmente, non funziona perchè, dopo l'uso, il dischetto ripresenta gli stessi file di prima.
(Mario Romoli - Firenze)

• Il programma serve proprio a quello! Forse l'articolo non era sufficientemente chiaro e ne approfitto per ricordare il suo uso e la sua funzione.

• Quando si formatta un dischetto (con la forma sintattica "breve", senza ID, altrimenti nessun recupero è possibile) la formattazione si limita ad alterare alcune tracce relative alla directory.

• Ripristinando le funzionalità della directory si riesce a "recuperare" tutti i file, ad eccezione dei primi otto, che sono stati alterati dalla formattazione di cui sopra.

• Se, quindi, facciamo in modo che ogni nostro dischetto abbia i primi otto file privi di qualsiasi importanza, avremo la certezza di poter recuperare SEMPRE i file dei nostri floppy, anche se accidentalmente formattati.

Il programma Antiformat, quindi, presenta un menu con tre opzioni: la prima chiede se si desidera "preparare" dischetti vergini (aggiungendo, cioè, otto file "inutili"); la seconda se si desidera aggiungere gli otto file di cui sopra a dischetti già riempiti, e la terza, infine, effettua l'operazione di salvataggio, ripristinando la directory, ad eccezione dei primi otto file.

Validate

☐ A che serve il comando Validate per il disk drive?
(Piero De Giorgio - Taranto)

• A volte capita, lavorando con il disk drive, di combinare pasticci, sia operando con file sequenziali o relativi (dimenticando di chiuderli dopo averli aperti) che con i programmi (tentando di registrare un programma che non può essere ospitato su disco a causa della modesta quantità di blocchi liberi, oppure assegnando nomi illegali ai programmi stessi).

Dopo tali operazioni la directory di un dischetto presenta alcuni nomi di file, preceduti da un asterisco, di lunghezza apparentemente nulla. Le presenze di questi nomi "strani" possono alterare il normale funzionamento del drive e, in ogni caso, occupano blocchi che potrebbero esser lasciati liberi per altri scopi.

A ripristinare le condizioni ottimali del floppy disk provvede, appunto, il comando validate, che non solo cancella i nomi dei file errati ma, soprattutto, rende nuovamente disponibili i blocchi da loro occupati.

Ricambi Mps-802

☐ Indispettito per la mancata riparazione della mia Mps-802, ho preferito cambiare totalmente rotta, acquistando un computer Ms-Dos compatibile di marca tassativamente NON Commodore.
(Giulio Carlucci - Brindisi)

• Lettere come questa, ovviamente, dispiacciono, e non poco.

Naturalmente il nostro lettore ha ragione di lamentarsi pur se, a nostro avviso, la "punizione" inflitta ci sembra eccessiva.

Il negoziante, da parte sua, avrebbe potuto fare qualche sforzo ritirando la stampante non più riparabile ed offrendone, a prezzo super scontato, una più moderna: forse non avrebbe guadagnato nulla (magari ci avrebbe rimesso qualcosa) ma non avrebbe perso un prezioso cliente.

Molti commercianti, purtroppo, non si rendono conto che un cliente ha sempre ragione, se non altro perchè, come minimo, si rivolge ad altri e non può fare a meno di sconsigliare, ai propri conoscenti, il nominativo del negozio dal quale ritiene di aver subito un trattamento poco corretto.

Linguaggio macchina

□ Molti lettori, il cui elenco sarebbe troppo lungo da riportare, desiderano "passare" al linguaggio macchina, ma hanno una gran confusione in testa. In questo riquadro vedremo di chiarire, almeno in parte, le idee.

• Naturalmente invitiamo i lettori interessati a leggere con la massima attenzione tutti gli articoli che Commodore Computer Club dedica all'argomento; nelle brevi note che seguono, infatti, siamo costretti ad una schematicità estrema.

• Tutti i computer, tra cui il C/64, il C/128, il C/16 ed il Plus/4, sono costituiti da un insieme di circuiti integrati che possono esser gestiti, a livello elettronico, esclusivamente per mezzo della matematica binaria (costituita dai due simboli 0 ed 1).

• Poichè tale notazione è notevolmente complessa per la maggior parte delle persone, sono stati realizzati linguaggi ad alto livello, tra cui il Basic, molto più semplici e di immediato apprendimento.

• Nonostante ciò, il computer continua ad essere una macchina binaria, continua a ragionare in binario e, in definitiva, continua ad operare in linguaggio macchina, l'unico lingauggio che conosce.

• Tutti i comandi e le istruzioni Basic digitati sulla tastiera, pertanto, vengono immediatamente convertiti in linguaggio macchina, per sottoporli alle operazioni del caso.

• Per verificare che, realmente, il computer (o meglio: il suo interprete Basic oppure il suo sistema operativo) converta i vari comandi in byte, è possibile ricorrere all'istruzione Peek che consente di conoscere il contenuto di qualunque locazione di memoria (detti anche: byte, cella).

• Per alterare il contenuto di una qualunque locazione di memoria, al contrario, è necessario fare uso del comando Poke A,B in cui "A" rappresenta l'indirizzo della cella che si intende modificare e "B" è il valore che si desidera trascrivere.

• Mentre, però, l'operazione Peek non altera mai il contenuto di una locazione, il comando Poke, proprio per l'azione che svolge, si può rivelare molto pericoloso perchè, se mal adoperato, rischia di bloccare il computer.

• Conoscendo il significato dei codici in linguaggio macchina (oppure limitandosi a trascrivere, con la massima attenzione, i programmi cosiddetti "caricatori") è possibile, grazie alle istruzioni Read...Data (ed utilizzando soltanto il Basic) trascrivere in zone di memoria opportune, alcuni programmi scritti in linguaggio macchina che, in seguito, devono essere attivati con la Sys opportuna.

• Spesso, soprattutto su alcune riviste, non vengono pubblicati i programmi caricatori, ma solo i listati in Assebly, che è un linguaggio, da procurarsi a parte, di cui esistono numerose versioni, ma tutte destinate ad utenti abbastanza esperti.

• Riepilogando: un computer ragiona sempre ed esclusivamente in binario; è possibile caricare un programma l.m. mediante un listato Basic (che conterrà, pertanto, istruzioni Read...Data e Sys) a patto di seguire con la massima attenzione i consigli descritti nei vari articoli; è possibile comprendere i disassemblati commentati del linguaggio più evoluto del l.m. "puro" (e ci riferiamo all'Assembly) a patto di averlo studiato con una certa cura.

• Avete notato, comunque, che non solo abbiamo pubblicato un fascicolo speciale dedicato al l.m. ma, addirittura, dal n.50 stiamo trattando l'Assembly in maniera sistematica?

Naturalmente non è possibile recuperare il contenuto dei file registrati in modo errato, dal momento che un comando validate fa piazza pulita di ogni loro traccia.

Il comando validate, inoltre, non deve essere usato per nessun motivo con dischetti protetti e, in genere, commercializzati: a causa del suo funzionamento, infatti, si rischia di distruggere "errori" che non sono considerati tali da molte procedure di protezione. Effettuare un'innocente operazione di validate, pertanto, provoca, in questi casi, la distruzione di preziose informazioni con la conseguenza di non veder più funzionare il programma.

Come nasce un programma

□ Ho digitato il listato dell'inserto del N.50 ("Come nasce un programma") ma non riesco a capire che cosa rappresentano le cifre posizionate alla destra dei valori che appaiono sul video.

(Mario Tripodina - Palmi)

• Il listato "Terza miglioria" elabora i risultati da visualizzare nel senso che li divide in due parti: quella intera e quella decimale; come elemento di separazione considera il punto (cioè... la virgola) dopo aver trasformato, in stringa, il valore elaborato.

Ne consegue che le parti intere dei valori risultano perfettamente allineate; per ciò che riguarda, invece, le parti decimali, queste non sempre sono quelle che ci aspettiamo, a causa di arrotondamenti interni del computer.

L'elaborazione, tuttavia, non può tener conto di simili "preziosismi" e ne consegue che vengono stampati i valori decimali secondo la notazione esponenziale.

Per fare in modo che la visualizzazione sia corretta anche per la parte decimale, pertanto, è necessario introdurre una modifica che, tenendo conto dell'importanza della parte stessa, "decida" di stamparla oppure no.

FANTASTICO **Il primo vero poker per C 64**

N.17 - Lire 8.000 ***Il computer gioca per tre, tu sei il quarto***

Software Club

C 64/128

- Cover • Outrider • Poker
- Guerrilla • Galaxy Cargo
- Skema • Graphic Manager

C 16/+4

- Cover • Fury • Frog

Galaxy Cargo

Outrider

Guerrilla

systems

Commodore Club - Dir. Resp.
A. Ronchetti Edizioni Systems
Editoriale Srl - V.le Famagosta
75 -20142 Milano - Reg. Trib. MI.
n. 104 del 25/2/84 - Distr. MePe.

SE CI PORTI IL TUO VECCHIO COMPUTER,

1084
Commodore

ECCO I COMMODORE POINT CHE CAMBIANO IL TUO VECCHIO COMPUTER CON IL PRINCIPE:

Commodore

Lombardia

AL RISPARMIO di ADA CASTOLDI e SALA S.
V.le Monza, 204 - MILANO

BRAHA ALBERTO s.d.f.
Via Pier Capponi, 5 - MILANO

E.D.S. ELECTRONIC DATA SYSTEMS s.r.l.
C.so Porta Ticinese, 4 - MILANO

E.S.C. s.r.l.
Via Roggia Scagna, 7 - MILANO

FAREF S.p.A.
Via A. Volta, 21 - MILANO

GBC ITALIANA S.p.A.
Via Petrella, 6 - MILANO
Via Cantoni, 7 - MILANO
V.le Matteotti, 66 - CINISELLO BALSAMO

GIGLIONI LAURA
Via D'Ovidio, 8 - MILANO

GIGLIONI s.r.l.
V.le L. Sturzo, 45 - MILANO

LOGITEK s.r.l.
Via Golgi, 60 - MILANO

MARCUCCI S.p.A.
Via F.lli Bronzetti, 37 - MILANO

MELCHIONI S.p.A.
Via P. Colletta, 37 - MILANO

MESSAGGERIE MUSICALI S.p.A.
Galleria del Corso - MILANO

NEWEL s.r.l.
Via Mac Mahon, 75 - MILANO

RIVOLA s.n.c.
Via Vitruvio, 43 - MILANO

F.LLI GALIMBERTI s.a.s.
Via Nazionale dei Giovi, 28/36 BARLASSINA (MI)

CASA DELLA MUSICA s.a.s.
Via Indipendenza, 21 - COLOGNO MONZESE (MI)

PENATI s.r.l.
Via Verdi, 28/30 - CORBETTA (MI)

EPM System
V.le Italia, 12 - CORSICO (MI)

P. GIORGIO OSTELLARI
Via Milano, 300 - DESIO (MI)

CENTRO COMPUTER PANDOLFI
Via Corridoni, 18
LEGNANO (MI)

COMPUTER s.a.s. di DE ANDREIS & C.
Via Vecellio, 41 - LISSONE (MI)

FUTURA s.a.s.
Via Solferino, 31 - LODI (MI)

MBM - INFORMATICA SYSTEMS S.p.A.
C.so Roma, 112 - LODI (MI)

L'AMICO DEL COMPUTER s.a.s.
V.le Lombardia, 17
MELEGNANO (MI)

BIT 84 s.a.s.
Via Italia, 4 - MONZA (MI)

I.C.O. di Giorgio Odorici
Via dei Tigli, 14 - OPERA (MI)

COMIF
Via Autolinee, 10 - BERGAMO

CORDANI S.p.A.
Via dei Caniana, 8 - BERGAMO

D.R.B. di De Ruschi Ernesto
Via Borgo Palazzo, 6 - BERGAMO

NEW SYSTEMS di Mazza Carlo R. & C.
Via Paglia, 36 - BERGAMO

COMPUTER TEAM hi-tec s.r.l.
Via Verdi, 1/B - CARVICO (BG)

OTTICO OPTOMETRISTA ROVETTA di Vasco Vasconi
P.zza Garibaldi, 6 - LOVERE (BG)

A.I.S. International s.r.l.
Via San Carlo, 25 - SAN PELLEGRINO TERME (BG)

SISTHEMA s.r.l.
Via Roma, 45 - SARNICO (BG)

COMPUTER CENTER
Via Cipro, 62 - BRESCIA

INFORMATICA 2000
Via Stazione, 16/B - BRESCIA

VIGASIO MARIO
Portici Zanardelli, 3 - BRESCIA

MISTER BIT
Via Mazzini, 70 - BRENO (BS)

CAVALLI PIETRO
Via 10 Giornate, 14B
CASTREZZATO (BS)

VIETTI GIUSEPPE
Via Milano, 1/B - CHIARI (BS)

MEGABYTE di Trabucchi
P.zza Duomo, 17 - DESENZANO DEL GARDA (BS)

DITTA BARESI RINO & C. s.n.c.
Via XX Settembre, 7 - GHEDI (BS)

INFO CAM s.r.l.
Provinciale 38
GRATACASOLO (BS)

KOFBAKER s.r.l.
Via Marchi, 65/B
VESCOVATO (BS)

2M ELETTRONICA s.r.l.
Via Sacco, 3 - COMO

IL COMPUTER di Ferrari Catia
Via Indipendenza, 90 - COMO

ELTRONGROS S.p.A.
Via L. Da Vinci, 54
BARZANÒ (CO)

EGA di Giuseppe Abello
Via Mazzini, 42
CASSAGO BRIANZA (CO)
Filiale: Via Aldo Moro, 17
GALBIATE (CO)

DATA FOUND computer shop
Via A. Volta, 4 - ERBA (CO)

CIMA ELETTRONICA s.a.s.
Via Leonardo da Vinci, 7 - LECCO (CO)

FUMAGALLI
Via Cairoli, 48 - LECCO (CO)

RIGHI ELETTRONICA
Via G. Leopardi, 26
OLGIATE COMASCO (CO)

MONDO COMPUTER
Via Giuseppina, 11/B - CREMONA

TELCO di Gianfranco Zambiasi & C. s.n.c.
Piazza Marconi, 2/A - CREMONA

PRISMA s.n.c.
Via Buoso da Dovara, 8
CREMONA

ELCOM GBC
Via IV Novembre, 56/58
CREMA (CR)

EUROELETTRONICA
Via XX Settembre
CREMA (CR)

COMPUTER s.a.s. di Tubaldo Elso & C.
Galleria Fermi, 7 - MANTOVA

ELETTRONICA DI BASSO
Viale Risorgimento, 69
MANTOVA

32 BIT (Computer Studio)
Via Cesare Battisti, 14
MANTOVA

POLIWARE s.r.l.
Corso Carlo Alberto, 66 - PAVIA

LOGICA INFORMATICA s.r.l. computer shop
V.le Monte Grappa, 32
VIGEVANO (PV)

M. VISENTIN
C.so Vittorio Emanuele, 76
VIGEVANO (PV)

COMPUTER LINE
Via G. Carducci, 4 - PIACENZA

DELTA COMPUTER
Via Martiri della Resistenza, 15/4
PIACENZA

SOVER s.n.c.
Via IV Novembre, 60
PIACENZA

CIPOLLA MAURO
Via Tremogge, 25 - SONDRIO

FOTONOVA di Bianchi Rita & C. s.n.c.
SAN PIETRO DI BERBENNO (SO)

DIMECO SISTEMI s.n.c.
Viale Garibaldi ang. Iva
VARESE

IL CENTRO ELETTRONICO s.r.l.
Via Morazzone, 2 - VARESE

SUPERGAMES s.a.s. di Ranzoni Franco
Via Carrobbio, 13 - VARESE

BUSTO BIT di Vittorio Ornago
Via Gavinana, 17 - BUSTO ARSIZIO (VA)

CRESPI GIUSEPPE & C. s.n.c.
V.le Lombardia, 59
CASTELLANZA (VA)

COMPUTER SHOP s.n.c.
Via A. Da Brescia, 2
GALLARATE (VA)

LIMA IMPORT EXPORT s.r.l.
c/o Grandi Magazzini Bossi
Via Clerici, 196
GERENZANO (VA)

I.A.C. nuove tecnologie s.n.c.
Via Matteotti, 38
SESTO CALENDE (VA)

Piemonte

ABA ELETTRONICA s.n.c.
Via C. Fossati, 5/P - TORINO

ALEX COMPUTER E GIOCHI di Bovolenta Diana
C.so Francia, 333/4 - TORINO

COMPUTER HOME
Via San Donato, 46/B - TORINO

COMPUTING NEW s.n.c. di Chiricosta Paolo & C.
Via Marco Polo, 40/E - TORINO

DE BUG di Guglielmotto Roberto
C.so Vittorio Emanuele II, 22
TORINO

DESME UNIVERSAL s.a.s.
Via San Secondo, 95 - TORINO

F.D.S. ALTERIO di Fernando De Siati
Via Borgaro, 86/D - TORINO

INFORMATICA ITALIA s.r.l.
C.so re Umberto, 129 - TORINO

MT INFORMATICA s.r.l.
C.so Giulio Cesare, 58 - TORINO

NEW BUSINESS COMPUTER s.a.s.
Via Nizza, 45 - TORINO

RADIO TV MIRAFIORI di Maggiani e Barile s.n.c.
C.so Unione Sovietica, 381
TORINO

SMT ELETTRONICA s.n.c. di S. Patrucco & C.
Via Bibiana, 83/B - TORINO

PAUL E CHICO VIDEOSOUND di Varetto F. & C. s.n.c.
Via Vittorio Emanuele, 52
CHIERI (TO)

BIT INFORMATICA
Via V. Emanuele, 154
CIRIÈ (TO)

HI-FI CLUB
C.so Francia, 92/C
COLLEGNO (TO)

I.C.S. s.r.l.
Stradale Torino, 73
IVREA (TO)

CERUTTI s.n.c.
Corso Torino, 234
PINEROLO (TO)

EUREX di Ponchia Rag. Gianfranco & C. s.a.s.
C.so Indipendenza, 5
RIVAROLO C.SE (TO)

BIT MICRO di Gras Ginette
Via Mazzini, 102 - ALESSANDRIA

WEST RECORDS (soc. Servizi Informatici S.a.s.)
C.so Roma, 85 - ALESSANDRIA

S.G.E. ELETTRONICA di Solarolo Giuseppe
Via Bandello, 19 - TORTONA (AL)

RECORD di Fiorina Giuseppe
C.so Alfieri, 166/3
(Galleria Argenta) - ASTI

ROSSI COMPUTERS di Rossi Metello & C.
C.so Nizza, 42 - CUNEO

PUNTO BIT s.n.c. di Vacca Giovanni & C.
C.so Langhe, 26/C - ALBA (CN)

SDI s.n.c.
Via Vittorio Emanuele, 250
BRA (CN)

CURETTI AUGUSTO
C.so Italia, 3 - Mondovì (CN)

ELCOM s.r.l.
C.so Mazzini, 11 - NOVARA

PROGRAMMA 3 s.n.c.
V.le Buonarroti, 8 - NOVARA

PUNTO VIDEO di La Vecchia Domenico
C.so Risorgimento, 391
NOVARA

COMPUTER di Mirco Polacco & C. s.a.s.
Via Monte Zeda, 4 - ARONA (NO)

ALL COMPUTER di Peirotti Alberto
C.so Garibaldi, 106
BORGOMANERO (NO)

MICROLOGIC s.a.s.
Via Giovanni XXIII, 2
DOMODOSSOLA (NO)

ELLIOTT COMPUTER SHOP s.r.l.
Via Don Minzoni, 32
INTRA (NO)

DITTA ELETTROGAMMA s.a.s.
C.so Bormida ang. Via Montanara
VERCELLI

ELETTRONICA s.a.s. di Bellamo A. & C.
Strada Torino, 15 - VERCELLI

C.S.I. TEOREMA s.r.l.
Via Losana, 9 - BIELLA (VC)

FOTOSTUDIO TREVISAN
Via XXV Aprile, 24/B
CORSATO (VC)

STUDIO FOTOGRAFICO IMARISIO di Imarisio Roberto
P.zza Martiri Libertà, 7
TRINO (VC)

Redo From Start

□ Che significato ha il messaggio di errore che compare spesso in fase di Input?
(Maurizio Colò - Roma)

• "Redo from start" significa, letteralmente: "Rifai dall'inizio" e viene emesso solo in fase di Input.

Per comprenderne la funzione, prova a digitare il seguente mini-programma:

```
100 Input A
110 Print A
120 Goto 100
```

Non fa altro che chiedere un numero, associarlo alla variabile A, stamparla e ripartire daccapo.

Se vengono battuti solo caratteri numerici, tutto funziona regolarmente; se, invece, vengono premuti tasti alfabetici, compare il messaggio citato.

Si noti che, a differenza di altri tipi di errore, questo non blocca l'esecuzione del programma: tutt'al più "sporca" l'eventuale maschera presente, in quel momento, sullo schermo.

Il motivo di tale comparsa è dovuto al fatto che il computer tenta di associare alla variabile numerica "A" una stringa alfabetica che, come è noto, può solo essere associata ad una variabile stringa (come, ad esempio, A$).

Volendo evitare la comparsa del messaggio, pertanto, è necessario apportare una piccola modifica:

```
100 Input A$
115 A=val(A$)
110 Print A
120 Goto 100
```

In questo modo, qualunque sia la sequenza di carateri battuta (ad eccezione del doppio punto e della virgola) ne verrà calcolato il valore che, nel caso di caratteri alfabetici, verrà posto a zero, ma non permetterà l'emissione del fastidioso messaggio.

H-copy senza spaziature

□ Vorrei utilizzare una routine di h/copy per la mia Mps-802 che non lasci alcuno spazio vuoto tra una schermata e l'altra. Tale opportunità mi consentirebbe di ottenere, nel caso di diagrammi molto lunghi, un'unica striscia stampata.
(Giampaolo Guarnieri - Lugo)

• Tutte le routine di hard-copy, di solito, effettuano almeno un return "a vuoto", dopo aver stampato l'ultimo rigo, per evitare fastidiose ed antiestetiche sovrapposizioni di caratteri.

Purtroppo non posseggo da tempo la Mps-802 (ma solo una compatibile 803) e non me la sento di pubblicare una routine senza averla prima provata.

Sul fascicolo "Commodore speciale" (interamente dedicato al linguaggio macchina ed alle routine grafiche) è presente, a pag. 75, una mini-micro routine specifica per Mps-802 che, opportunamente modificata per ciò che riguarda gli indirizzi della pagina grafica utilizzata, dovrebbe servire allo scopo.

Club

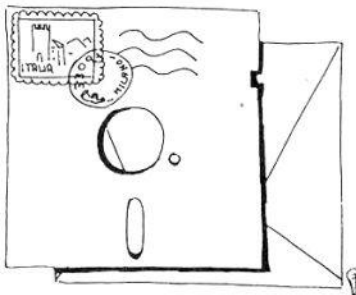
□ Ho alcune idee per aprire un Super-Club C/64, da gestire per corrispondenza a livello nazionale, che vorrei sottoporre alla vostra attenzione (segue elenco).
(Massimiliano Rosso - Aosta)

• Le idee sono buone, anzi ottime ma, come al solito, tra il dire e il fare c'è di mezzo il mare delle difficoltà pratiche.

Per gestire seriamente un super club, del tipo ipotizzato dal nostro lettore, c'è bisogno di tanto tempo, pazienza e... quattrini!

Il minimo che si possa fare, infatti, è quello di raggiungere i potenziali aderenti (a proposito: come procurarsi i loro indirizzi?) per mezzo di un fascicoletto illustrato; siccome, per garantirsi un minimo successo, sono necessari almeno 2000 nominativi, puoi immaginare facilmente le prime spese di pubblicazione ed affrancatura.

In ogni caso, poi, il giornalino del club (di pubblicazione almeno mensile, e non trimestrale, come tu suggerisci: l'informatica corre!) deve offrire, ad un prezzo concorrenziale, informazioni e "servizi" migliori di quelli offerti dalle riviste del settore che di solito si trovano in edicola, altrimenti non converrebbe procurarselo. Gli autori degli articoli del giornalino, inoltre, dovrebbero esser pagati meglio di quanto possano offrire le case editrici, altrimenti...

Purtroppo la realtà commerciale di un qualsiasi fenomeno è piuttosto dura e difficile da gestire. Noi della redazione, più di una volta, abbiamo pensato di realizzare un club del tipo suggerito, pensando soprattutto al vantaggio notevole (in termini di pubblicità gratuita) che ci viene offerto da questa stessa rivista; abbiamo tuttavia sempre desistito dall'impresa (che costerebbe alcuni milioncini, solo per iniziare!), preferendo dedicare a Commodore Computer Club tutte le risorse e le energie che, altrimenti, si disperderebbero in troppe attività.

H-Copy non copia

☐ Ho usato la funzione Append tra il programma Koala ed una vostra routine di hard copy per Mps-802, ma la procedura non funiona. Come mai?
(Mario Morniroli - Arma di Taggia)

• Il programma Append (da noi pubblicato più di una volta, in varie occasioni) può funzionare solo se i due programmi da legare tra loro sono scritti in Basic. Koala, invece, è scritto in l.m. (bechè sembri costituito da una sola istruzione del tipo: Sys...) ed una fusione è impossibile.

Abbiamo già pubblicato varie versioni di hard-copy (ed altre ne pubblicheremo) per Mps-803; la 802, infatti, è fuori produzione da molto tempo e non verranno più trattati argomenti ad essa legati. Molti utenti che non volevano disfarsene, se non bastasse, hanno provveduto a sostituire la sua Rom, dal prezzo irrisorio, che la rende compatibile con i programmi Mps-803.

Senza risposta

☐ Perchè avete condensato la domanda, presente nella mia lettera, apparsa di recente sulla vostra rivista? Se avete coraggio, pubblicatela per intero...
(Ermanno Filipponio - Lecco)

• Tutte le lettere vengono condensate: tranne casi estremamente particolari, tale atteggiamento è seguito

da tutte le pubblicazioni di questo mondo.

Non appena una lettera, condensata o meno, viene pubblicata, viene anche immediatamente cestinata; e questo non certo per disprezzo, ma per una comprensibile mancanza di spazio per l'archiviazione.

Invece di far polemica, pertanto, era meglio riproporre la domanda (alla quale avrei risposto certamente, visto l'atteggiamento piuttosto duro del nostro lettore), invece di riempire la lettera di insulti più o meno gratuiti.

Rette in hi-res

☐ Come posso tracciare, con il mio C/64, rette in alta risoluzione?
(Michele Loconsole - Bari)

• Abbiamo più volte affrontato l'argomento, ed ancora vi ritorneremo; al momento attuale delle cose, tuttavia, ti consiglio di procurarti uno dei numerosi Tool che, grazie a comandi Basic aggiuntivi, permettono di disegnare in hi-res con la massima facilità. Mi permetto di suggerirti il nostro emulatore Gw-Basic, su dischetto, che puoi ancora trovare in edicola.

Quanta memoria?

☐ Ho digitato il vostro programma "Espansione di memoria per C/16" ma non sempre la quantità di memoria libera è la stessa.
(Alberto Zazza - Beduzzo)

• Il programma cui si riferisce il nostro lettore consente di assegnare al Basic una parte della zona del video spostando i puntatori di inizio ed approfittando del fatto che la memoria video si trova immediatamente "prima" dell'area destinata al Basic.

Naturalmente il programma non può immaginare che cosa ci sia "dopo". Se, in altre parole, vengono alterati anche i puntatori di fine Basic (per allocare, supponiamo, una routine l.m.) è logico che la quantità di memoria che rimane a disposizione diminuisce di conseguenza.

BBS Amiga Users

E' nato il BBS Amiga Users: opera con Amiga 2000 ed è attivo, tutti i giorni, dalle 22:30 alle 7:00. Dati tecnici per entrare in contatto: Amiga BBS tel.051/23.54.92 Sysop d.j. byte 300/1200 baud, parametri 8/n/1 ore 22:30 - 7:00

UN'EMOZIONE DA 1200 BIT AL SECONDO

- La potenza di una banca dati, la dinamica di un quotidiano.
- L'unico servizio telematico italiano con le notizie in tempo reale sul mondo dell'informatica.
- Il solo accessibile tramite la rete nazionale Videotel presente in più di 67 distretti telefonici (oltre 1000 comuni!).
- Con LASERNET 800 potrai caricare programmi in TELESOFTWARE, chiacchierare in diretta con tutta Italia sulle CHATLINES, editare un tuo spazio personale su PRIMA PAGINA, leggere le notizie più interessanti di LASER NEWS e migliorare la tua programmazione con i nostri corsi.
- Oltre 5000 pagine consultabili 24 ore su 24.
- Il nostro servizio ti costa ogni giorno meno della metà di un quotidiano!

... ... PROVALA!

Per avere maggiori informazioni sul servizio compila il tagliando e spediscilo a:
LASERNET 800 - Via G.Modena, 9
20129 Milano - Tel. 02/200.201

Desidero ricevere maggiori informazioni su LASERNET 800

Cognome.......................... Nome..........................
Via..
Città.. Prov.............
CAP.................................. Tel..........................
Data di nascita................/......./.................
Il mio computer é un:

Commodore	☐ 64	☐ 128	☐ Amiga
☐ MSX	☐ BBC	☐ Atari ST	☐ PC
Spectrum	☐ 48K	☐ Plus	☐ 128

☐ Ho già un adattatore telematico

Tot Corrector

□ Il signor D. Di Mario di Milano, ci scrive congratulandosi a proposito dello "Speciale Totocalcio", ma anche per segnalare un paio di dubbi sul funzionamento del programma "Tot Corrector", contenuto nel fascicolo menzionato.

• Il primo appunto riguarda la funzionalità delle riduzioni eseguite dal programma.

Benchè questo argomento sia già stato trattato nel numero 41 di "Commodore Computer Club", cercheremo di sintetizzarne il contenuto, per coloro che non fossero in possesso del fascicolo citato.

I sistemi ridotti mirano alla individuazione delle colonne di un sistema che possano garantire lo scarto di un solo punto con quelle non prese in considerazione.

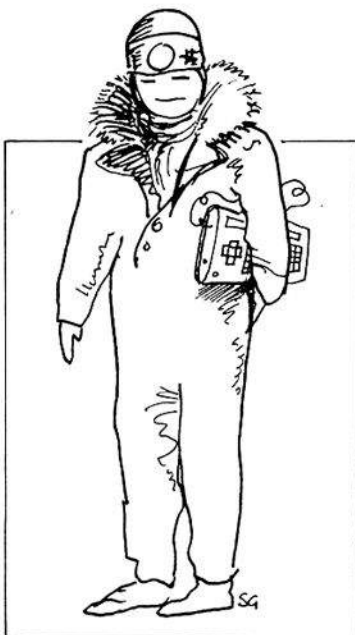
Ciò vuol dire che se, in un sistema formato da 5 doppie (integrale = 32 colonne), la riduzione desiderata seleziona solo quattro colonne, queste devono essere in grado di garantire che, nel caso in cui la colonna vincente appartenga a quelle eliminate, si realizzi, tuttavia, almeno un 12 (caso del Totocalcio).

Purtroppo le colonne selezionate non hanno una disposizione regolare all'interno del sistema che le genera, rendendo quanto mai ardua l'elaborazione di un algoritmo matematico per la loro individuazione.

In ogni caso il programma è destinato non già alla riduzione, quanto alla "correzione" d'errore.

Il difficile procedimento di riduzione è inoltre ostacolato dal fatto che esso opera su di un sistema alterato da altre condizioni (correzione d'errore).

Dal momento che esistono diversi metodi per ottimizzare le giocate, bisogna saper scegliere il più appropriato, rispetto alle esigenze personali.

Chi gioca piccoli sistemi, magari da solo, dovrebbe orientarsi verso i sistemi ridotti, ottenendo un discreto vantaggio economico giocando

poche colonne, pur mantenendo pressochè inalterate le probabilità di vincita del sistema originario.

D'altro canto i giocatori più audaci, o coloro che giocano in società, disponendo di cifre più cospicue, potranno avvantaggiarsi delle possibilità offerte dai sistemi condizionati, a correzione d'errore ecc.

Per chiarire maggiormente il problema, consideriamo i seguenti sistemi:

A	B
1	1
1X	1X2
1	1
1X	X
X	1X2
1	1X2
1X	X2
1	1
X	X
1	1X
X	1X
X	1X2
1	1X2

Volendo selezionare le colonne che hanno un numero di segni "X" da 3 a 6, nel primo sistema eliminiamo solo una colonna:

1 X 1 X X 1 X 1 X 1 X X 1

cioè il 12,5% delle colonne totali, nel secondo eliminiamo 350 colonne, pari al 18% del sistema integrale.

Ecco spiegato perchè, condizionando piccoli sistemi (magari con riduzione), si ottiene un numero di colonne superiori alla sola riduzione, effettuata con il programma dedicato.

Per ciò che riguarda la seconda segnalazione (sempre per il programma "Tot Corrector"), invece, il nostro lettore ha perfettamente ragione!

Quando, nell'usare il programma citato, si selezionano alcune correzioni, facendo poi sviluppare il sistema, se si intende elaborarne uno nuovo, o modificare le correzioni del precedente, ci si imbatte in un problema: l'elenco delle correzioni viene cancellato ad eccezione della correzione di 0 errori.

La conseguenza di ciò consiste nel fatto che se (nella elaborzione precedente) era stata richiesta la correzione di 0 errori, questa rimane invariata, e viene quindi eseguita, in seguito, anche se non richiesta.

Il problema è risolvibile molto semplicemente modificando la sola linea 5660 del programma "Tot corrector V 1.0", a pagina 82, come di seguito riportata:

```
5660 For j=0 to 16: cr(j)=0: next
```

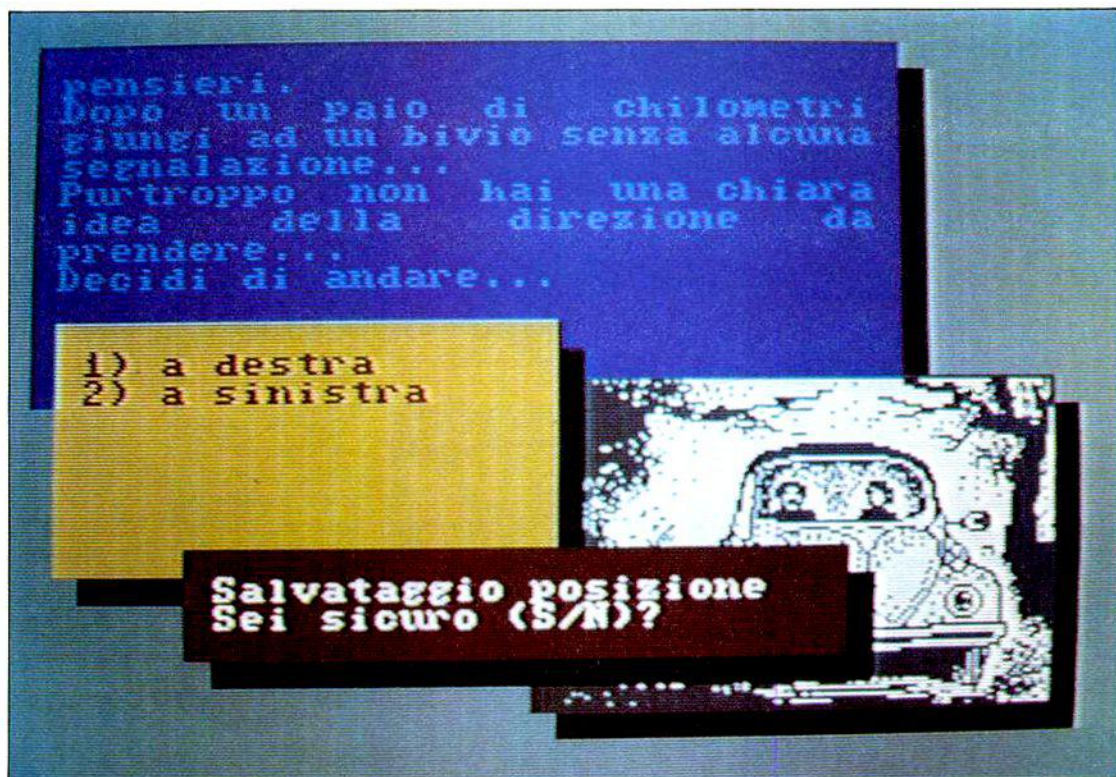
Per rassicurare i lettori, è d'obbligo precisare che il "bug" segnalato è di rilevanza marginale, dal momento che, in un qualsiasi sistema, esiste sempre una ed una sola colonna con 0 errori (cioè la colonna-base).

Di conseguenza il bug non comporta assolutamente errori di calcolo che possano pregiudicare la funzionalità del sistema stesso; questo inserisce tutt'al più, anche se non indicato, la colonna-base: in sostanza si gioca una colonna in più rispetto al dovuto, ma solo se, lo ripetiamo, il programma viene fatto girare più di una volta, e nelle condizioni particolari descritte!

(Antonio Pastorelli)

pensieri.
Dopo un paio di chilometri
giungi ad un bivio senza alcuna
segnalazione...
Purtroppo non hai una chiara
idea della direzione da
prendere...
Decidi di andare...

1) a destra
2) a sinistra

Salvataggio posizione
Sei sicuro (S/N)?

UN DISCHETTO DA INCUBO

Dylan Dog, il popolare personaggio dei fumetti della Daim Press, è ora su disco per un progetto destinato a ripetere e a superare il successo di Zagor

Il dischetto Commodore 64 Club n. 3, pubblicato qualche mese fa, conteneva, oltre a vari videogame ed utility, anche un'avventura ispirata a Zagor, uno dei personaggi dei fumetti più seguiti ed amati.

Il notevole successo riscosso dalla pubblicazione ha spinto la Systems Editoriale, di concerto con la Daim Press, a varare un'iniziativa senza precedenti nel panorama italiano.

Si tratta di una serie di dischetti che avranno per protagonista Dylan Dog, il popolare "Indagatore dell'incubo" che, nel suo genere, fa la parte del leone in tutte le edicole.

Un momento del gioco; le immagini visualizzabili sono numerosissime

Come per Zagor, si tratterà di giochi di ruolo, ovvero di giochi in cui il ragionamento è privilegiato rispetto all'abilità nell'uso del joystick. Il giocatore, coinvolto in oscure vicende, dovrà prendere le decisioni che riterrà più appropriate in funzione di ciò che gli viene suggerito dalle immagini, dalle descrizioni e perfino dagli effetti sonori.

Il colloquio, o meglio l'"interfaccia-utente", risulta molto evoluto a di-

spetto della intrinseca semplicità; al posto del consueto "parser" che costringe il giocatore ad indovinare le parole-codice che il computer conosce, esiste un sistema di scelte guidate che permette al giocatore di scegliere tra una gamma di possibilità predefinite.

Con tale modo di operare, in sostanza, si avvantaggiano coloro che amano i giochi di ruolo ma che non simpatizzano con il "parser", ritenuto spesso causa di eccessiva lentezza durante lo svolgimento degli eventi.

Il primo episodio, intitolato "Le notti della luna piena", è ispirato all'omonimo albo e vede Dylan Dog, accompagnato dal fedele Groucho, alle prese con misteriosi avvenimenti in un college femminile che, guarda caso, è situato nel profondo della foresta nera, terra quantomai infestata da licantropi, streghe e altre piacevolezze simili.

La copertina originale da cui è tratta l'ispirazione per il primo numero di Dylan Dog su | dischetto

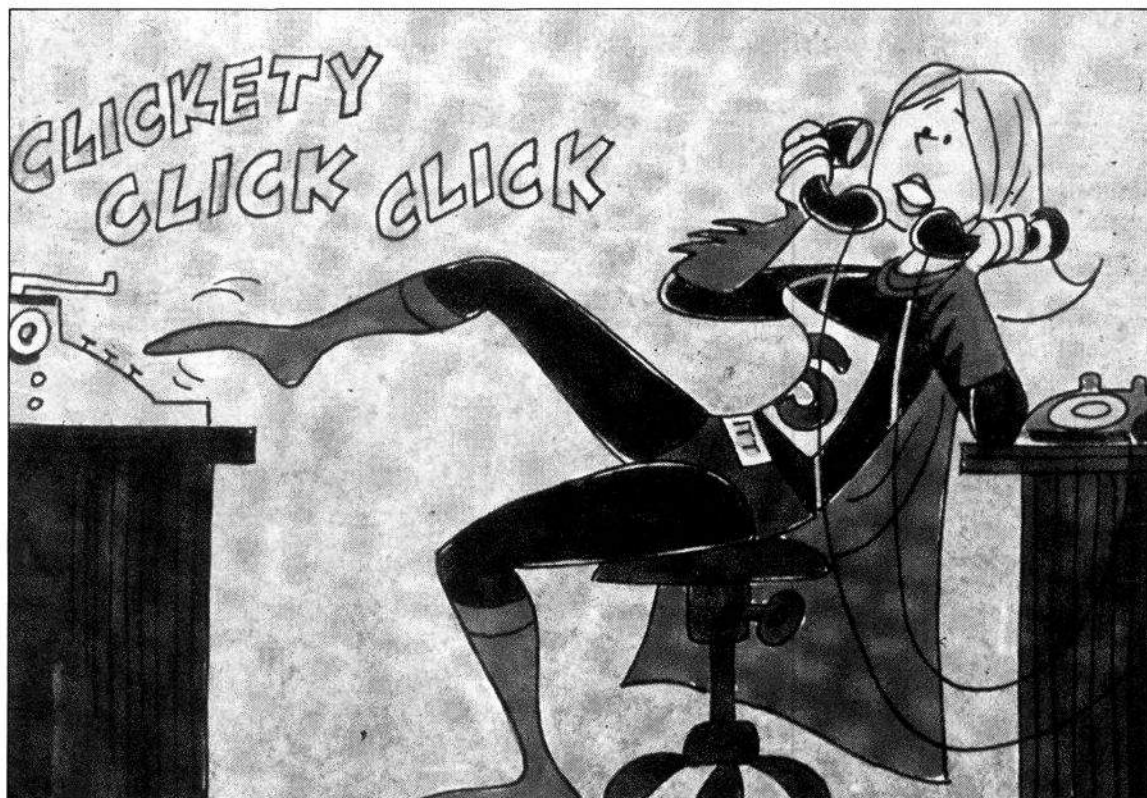
LA TECNICA

Ai più curiosi diremo soltanto che la struttura del package è stata dapprima progettata a tavolino, quindi sviluppata in ambiente MS-DOS, prendendo spunto da procedure in linguaggio C.

L'intera avventura è gestita tramite finestre in pagina grafica che permettono di visualizzare separatamente diverse decine di immagini in alta risoluzione.

Per la realizzazione della parte grafica si è ricorso all'intervento di un grafico professionista.

Gli effetti sonori, presenti per l'intera durata del gioco, sono ulteriori elementi di novità del prodotto, e tutto a vantaggio di un maggior coinvolgimento del giocatore.

In questo primo numero, oltre a "Le notti della luna piena", sarà presente anche un videogame "da bar", sempre ispirato a Dylan Dog, che grazie alla consueta e collaudata struttura garantisce divertimento anche al pubblico più giovane.

SCRIPTOR: UN WORD PROCESSOR PER C/16

Anche il piccolo computer Commodore può trasformarsi in un efficiente sistema di videoscrittura

di **Roberto Ferro**

Il listato di queste pagine si presenta tanto voluminoso da scoraggiare i più temerari che volessero digitarlo.

Per quanto riguarda la sua affidabilità vi possiamo assicurare che è stato esaminato attentamente e funziona, sempre che voi non commettiate banali errori di trascrizione. Per evitare dubbi, è stato fatto di tutto per rendere il listato facilmente leggibile tramite l'eliminazione di caratteri semigrafici e di controllo.

Se non possedete la stampante, praticamente vitale per programmi di w/p, diciamo subito che il programma è stato pensato per creare file compatibili con i più diffusi word processor per C/64, tra cui il favoloso "Easy Script" sicuramente noto a quel nostro amico che possiede C/64 e stampante.

SCHEDA TECNICA

Software per applicazioni "professionali"

Hardware richiesto: C/16, Plus/4; programma adattabile, con qualche difficoltà, anche al C/128.

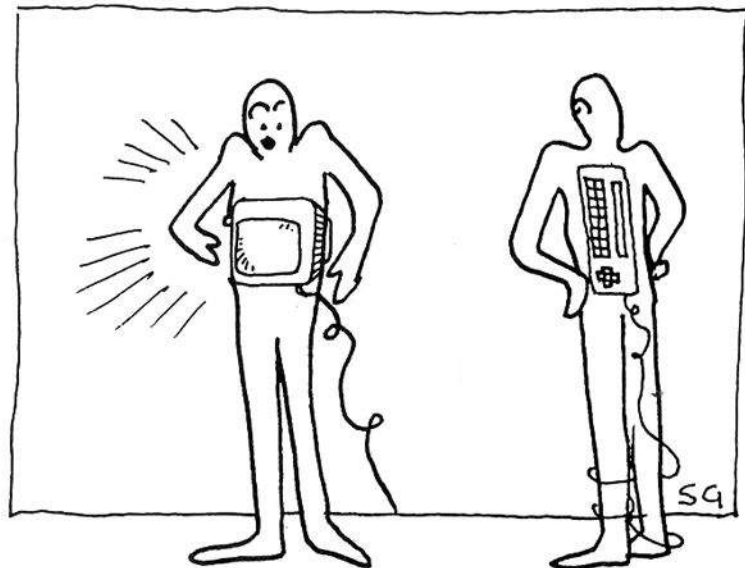
Ideale l'uso di una stampante e di un drive.

Anche il programma pubblicato in queste pagine è contenuto nel disco "Directory" di questo mese.

Scambiando il supporto magnetico, sarà quindi possibile avere i propri testi stampati. In poche parole questo significa che un testo da voi creato con SCRIPTOR potrà essere caricato, modificato e stampato con qualunque altro word processor che tratti file sequenziali. Che ve ne pare?

La compatibilità, tuttavia, è tale solo se i file (che sono in formato sequenziale "seq") vengono salvati su floppy disk; per il nastro magnetico, come è noto, viene usato un protocollo diverso nel caso di C/16 e C/64.

Appena lanciato, il programma provvede ad alterare i colori di sfondo e quelli di primo piano, mostra una linea verde in alto con il nome del programma e dell'autore, e compare un cursore che permetterà di digitare il testo.

I tasti per l'editing sono stati mantenuti, per quanto possibile, uguali a quelli che normalmente usiamo per i programmi Basic. Pertanto il tasto di INST/DEL funge da backspace (premuto da solo), e da insert (in unione con Shift). Quest'ultimo, e lo Shift Lock, permettono di visualizzare le maiuscole. Anche il Return conserva intatta la sua funzione.

L'editazione del testo è resa pittosto agevole in quanto anche i tasti cursore funzionano come tali, e ciò significa che non siamo affatto legati ad un editing per linea, come invece avveniva nei vecchi (pessimi) word processor. Il vantaggio è notevole in quanto per la correzione di una parola scritta, ad esempio, cinque linee più in alto sarà sufficiente raggiungerla con il cursore ed effettuare la correzione desiderata in modo totalmente "trasparente" e chiaro.

Altre funzioni si possono ottenere tenendo premuto il tasto Control (CTRL) mentre si batte un altro tasto:

CTRL + I

Inserisce una linea di spazi nel testo prendendo come punto di riferimento la posizione del cursore: viene spostato di una linea in giù tutto il testo a partire dal punto in cui si trova il cursore.

CTRL + D

Cancella una linea del testo prendendo ancora, come punto di riferimento, il cursore: la linea in cui si trova quest'ultimo viene eliminata.

Un programma molto usato

Se realizzassimo un'indagine sulla categoria di programmi più utilizzati, scopriremmo, con molta probabilità, che il Word Processing è una delle applicazioni più diffuse, se non addirittura la più diffusa.

Ciò è dovuto al fatto che l'elaborazione dei testi presenta molteplici vantaggi: infatti un testo può essere dapprima digitato per intero, dopodichè si potrà provvedere alla eliminazione di eventuali errori e ad una corretta impostazione delle frasi. Inoltre è possibile memorizzarlo su supporto magnetico prima di mandarlo in stampa in modo da disporre, anche in futuro, dello stesso testo senza essere costretti a conservare voluminosi plichi cartacei.

Un esempio di come tale modo di operare possa essere vantaggioso lo possiamo avere considerando la stesura di una lettera: una volta impostata correttamente la posizione del mittente, del destinatario, la data o qualunque altra specifica, potremo salvare il "documento" ed avere sempre disponibile uno standard a cui riferirci per le nostre lettere (questo è il caso tipico delle domande di lavoro a più Enti: devono contenere tutte lo stesso "corpo" ma avere differenti intestazioni).

Ovviamente il word processor può risultare utile in tantissimi altri casi come ad esempio per mettere in ordine appunti di studio, o preparare un chiara ed ordinata relazione, in modo da utilizzare il nostro home computer anche per "cose" un po' meno ludiche.

CTRL + N

Cancella l'intero testo e fa ripartire il programma. E' intuitivo che bisogna porre molta cautela nell'usare questa opzione perchè il testo cancellato non potrà più essere richiamato.

I tasti funzione

Quando il programma viene lanciato, prima di poter disporre del cursore viene chiesta la periferica che intendiamo usare per salvare il testo. Dovremo rispondere "D" se disponiamo di un disk drive, oppure "T" se abbiamo il solo registratore a cassette. In base a questa scelta iniziale verrano compiute tutte le successive operazioni di Input/Output.

I tasti funzione F1, F2 ed F4 permettono di salvare, caricare e stampare il testo; più precisamente:

F1: Carica

Una volta richiamata questa opzione, verrà richiesto (solo per il disk drive) il nome del file da caricare; in seguito cominceranno le operazioni di caricamento. Vedremo allora il testo scorrere, lettera per lettera, nell'angolo in alto a sinistra del video: questo è un segno che il caricamento procede bene.

A fine operazione bisognerà attendere qualche secondo per riavere il cursore disponibile, ma... sul video non compare assolutamente nulla.

Non si tratta di un errore, ma semplicemente del fatto che il testo caricato è in memoria, e per vederlo dovremo spostarci in alto con il cursore, tutto qui.

F2: Salva

Anche con questa opzione viene chiesto il nome del file (se vogliamo annullare l'operazione dovremo impartire un Return a vuoto); poi comincia il salvataggio.

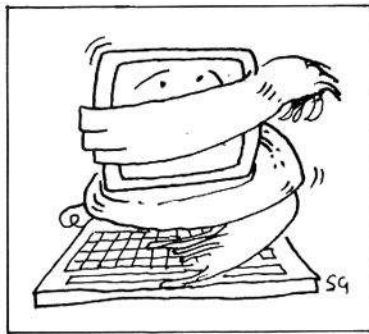
Per i possessori di disk drive è bene precisare che nonostante sia il Load che il Save prevedano un controllo sulla fattibilità dell'operazione, è meglio accertarsi che tutto sia in regola (drive acceso, disco ben inserito, levetta abbassata...) prima di compiere un salvataggio. Nel caso qualcosa non sia a posto il programma provvede ad annullare l'operazione, ad emettere un segnale sonoro nonchè ad illuminare il bordo dello schermo (nientepopodimeno!).

F4: Stampa

L'operazione di stampa è particolarmente semplice in quanto non richiede null'altro che la pressione del tasto in questione. Anche in questo caso è valido il discorso di prima: bisogna accertarsi che la stampante sia collegata ed accesa, soprattutto perchè in questo caso non è prevista nessuna forma di controllo.

Rimane infine da accennare al tasto F3. Esso è preposto all'attivazione e disattivazione del beep udibile quando si preme un tasto.

Prima di passare all'esame del listato è necessario illustrare alcune particolarità di SCRIPTOR.

Innanzitutto è possibile digitare fino ad un massimo di 152 linee (circa 6 schermate). Le operazioni di insert, backspace e return, attivate per "spezzare" una frase, sono consentite solo su paragrafi di cui sia visibile la fine sullo schermo, cioè il segno di return (carattere di freccia a sinistra).

Quando, infatti, compiamo una delle operazioni suddette, vedremo il cursore scattare alla ricerca della freccetta che indica il return, e solo se la raggiunge l'operazione sarà portata a termine.

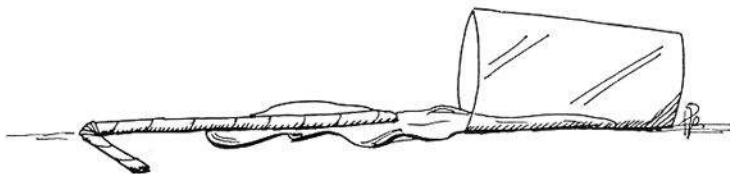
Nel caso in cui il cursore arrivi fino in fondo al video senza aver trovato nulla (il paragrafo termina quindi fuori dallo schermo), verrà emesso il segnale sonoro e l'illuminazione del bordo di cui sopra, per segnalarci che l'operazione è illegale e non può essere eseguita.

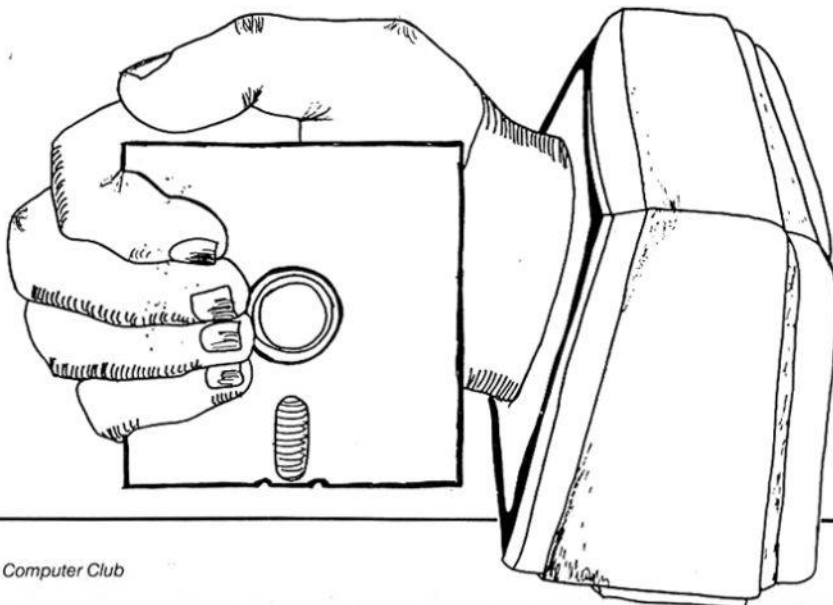
Note al listato

Come si può vedere tutte le porzioni di programma sono commentate, e pertanto eventuali migliorie o cambiamenti si possono fare rintracciando velocemente la zona in cui si intende operare. Avvertiamo sin d'ora che per il C/16 la quantità di memoria disponibile, una volta digitato il listato con tutte le REM, è piuttosto bassa, circa 500 bytes, e ciò è causato dal top di memoria che all'inizio del programma viene fissato a $2800. Da questa locazione in poi, infatti, viene memorizzato il testo.

Alle linee 230-240 viene verificato che il carattere immesso sia un carattere da scrivere oppure un comando. In quest'ultimo caso il controllo del programma passa alle linee 370 (e successive) dove viene identificato il carattere di comando e viene eseguito con un salto alla sezione di programma che se ne occupa.

Infine un cenno merita la routine di Save, dove risiede la parte di listato che trasforma il testo da codici Poke a codici ASCII (prima di salvarlo), in modo da mantenere la compatibilità, con la maggior parte degli altri word processor, di cui abbiamo parlato.

```
10 rem #################
20 rem
30 rem   scriptor v1.1
40 rem
50 rem #################
60 :
70 rem - di roberto ferro
80 :
90 poke55,0:poke56,40:fori=1to8:keyi,"":next
100 key1,chr$(133):key2,chr$(134):key3,chr$(135):key4,chr$(138)
110 color0,1,0:color1,6,6:color4,1
120 printchr$(14)+chr$(8):scnclr:vol8
130 printchr$(19)+chr$(19)+chr$(18)+" Scriptor v1.1        by  Roberto Ferro  "
140 poke2022,1:color1,2,6:printchr$(17)+"  (D)isk / (T)ape ?"
150 geta$
160 ifa$="d"thenpe=8:goto190
170 ifa$="t"thenpe=1:goto190
180 goto150
190 fori=32to95:le$=le$+chr$(i):next
200 fori=193to218:le$=le$+chr$(i):next
210 le$=le$+chr$(160):scnclr:in=10240:fi=16383:c=96:su=800:te=2:poke3112,224
220 geta$:ifa$=""then220
230 b=instr(le$,a$):a=asc(a$)
240 ifb=0then370:elsesc=sc+1:sound1,su,te
250 ifa>63anda<95thena=a-64:goto290
260 ifa>96anda<123thena=a-32:goto290
270 ifa>128thena=a-128
280 ifa=32thena=96
290 gosub1410:b=peek(3112+sc)
300 ifb=32thenb=96
310 ifc=31thenb=31:goto330
320 goto350
330 ifint(s/40)<>sc/40then350
340 poke205,int(sc/40)+1:gosub1090
350 poke3111+sc,a:poke3112+sc,b+128
360 c=b:goto220
370 ifa=17thenri=40:goto570
380 ifa=29thenri=1:goto570
390 ifa=157thenri=-1:goto570
400 ifa=145thenri=-40:goto570
410 ifa=20then680
420 ifa=148then840
430 ifa=13then940
440 ifa=134then1500
450 ifa=133then1690
460 ifa=135then530
470 ifa=3then1870
480 ifa=9then1920
490 ifa=14thenrun
500 ifa=138then2110
510 goto220
520 :
530 ifsu=800thensu=1021:te=0:elsesu=800:te=2:goto220
540 :
550 rem ** movimento cursore **
560 :
570 poke3112+sc,c
580 ifsc+ri<0andi1>0thenpoke205,int(sc/40)+1:gosub1090:gosub1280
590 ifsc+ri<0then630
600 ifsc+ri>959andi2>0thengosub1140
610 ifsc+ri>959then630
```

```
620 sc=sc+ri
630 b=peek(3112+sc):ifb=32thensc=sc-ri:b=c
640 poke3112+sc,b+128:c=b:goto220
650 :
660 rem ** backspace **
670 :
680 zx=1:goto1350
690 ifpeek(3111+sc)=32then750
700 ifsc=0then220
710 s=sc:poke3112+sc,c:sc=sc-1:poke3112+sc,c+128
720 dountilpeek(3112+s)=31orpeek(3112+s)=32:poke3112+s,peek(3113+s):s=s+1:loop
730 ifint(s/40)=s/40thenpoke205,int(s/40)+1:sys56992:gosub1160:goto220
740 poke3112+s,32:goto220
750 s=sc:sc=sc-1
760 dountilpeek(3112+sc)=31:sc=sc-1:loop:sa=sc
770 dountilpeek(3112+s)=32orpeek(3112+s)=31:poke(3112+sa),peek(3112+s)
780 poke3112+s,32:s=s+1:sa=sa+1:loop:poke3112+sa,peek(3112+s):poke3112+s,32
790 ifint(s/40)>int((sa)/40)thenpoke205,int(s/40)+1:sys56992:gosub1160
800 c=peek(3112+sc)-128:goto220
810 :
820 rem ** insert **
830 :
840 zx=2:goto1350
850 s=sc:poke3112+s,224
860 dountilpeek(3112+s)=31orpeek(3112+s)=32
870 b=peek(3113+s):poke3113+s,c:s=s+1
880 c=b:loop:d=s/40
890 ifint(d)=dthenpoke205,int(d)+1:gosub1090:poke3112+s,31:poke3152+s,c
900 c=96:goto220
910 :
920 rem ** return **
930 :
940 zx=3:goto1350
950 ifpeek(3113+sc)<>32andc<>31then1000
960 poke3112+sc,31:sc=int((sc/40)+1)*40
970 gosub1440
980 b=peek(3112+sc):ifb=32thenb=224:elseb=b+128
990 poke3112+sc,b:c=b-128:goto220
1000 ifi1+i2>152andpeek(4032)<>32thengosub1310:goto220
1010 ifpr>4031thenpr=0:gosub1310:goto220:elsed=40-(sc-int(sc/40)*40):s=sc
1020 dountilpeek(3112+s)=31orpeek(3112+s)=32:s=s+1:loop
1030 ifint(s/40)<int((s+d)/40)thenpoke205,int((s+d)/40)+1:gosub1090
1040 dountils+d+1=int(sc/40)*40+40:poke(3112+s+d),peek(3112+s):s=s-1:poke3113+s,
32:loop
1050 s=s+1:sc=s+d:poke3112+s,31:goto220
1060 :
1070 printchr$(19)+chr$(19):poke2022,1:return
1080 :
1090 ifpeek(4032)<>32thengosub1220
1100 sys55902:gosub1070:return
1110 :
1120 rem ** memorizza prima linea e mostra l'ultima se presente **
1130 :
1140 fori=0to39:pokein+i+(i1*40),peek(3112+i):next
1150 i1=i1+1:printchr$(27)+chr$(86):gosub1070
1160 ifi2=0thenreturn
1170 fori=0to39:poke4032+i,peek(fi+i-(i2*40)):next
1180 i2=i2-1:return
1190 :
1200 rem ** memorizza ultima linea **
1210 :
```

```
1220 fori=0to39:pokefi-40+i-(i2*40),peek(4032+i):next
1230 i2=i2+1:return
1240 :
1250 rem ** prende dalla memoria la prima linea **
1260 :
1270 gosub1070:printchr$(27)+chr$(87):gosub1070
1280 fori=0to39:poke3112+i,peek(in-40+i+(i1*40)):next
1290 i1=i1-1:return
1300 :
1310 sound1,900,10:color4,2:fori=0to50:next:color4,1:return
1320 :
1330 rem ** controlla che la fine del paragrafo sia visibile **
1340 :
1350 pr=3112+sc:pokepr,c:dountilpeek(pr)=31orpeek(pr)=32orpr>4070:pokepr,peek(pr
)+128
1360 pr=pr+1:pokepr-1,peek(pr-1)-128:loop:poke3112+sc,c+128
1370 ifpr>4070thengosub1310:goto220:elseonzxgoto690,850,950
1380 :
1390 rem ** controlla fine memoria **
1400 :
1410 ifsc<960thenreturn
1420 ifi1<152thengosub1120:sc=920:return
1430 sc=sc-1:poke3112+sc,a:return
1440 ifsc<960thenreturn
1450 ifi1+i2<152thengosub1120:sc=920:return
1460 sc=920:gosub1310:return
1470 :
1480 rem ** save **
1490 :
1500 h=1:i=0:op$="Save":gosub1980
1510 iflen(n$)=0then1650
1520 ifpe=1thenprintchr$(19)+chr$(19):poke2021,0:printchr$(19):is=1:goto1550
1530 is=5:open5,8,15,"i0:":close5
1540 ifds>0thengosub1310:goto1650:elsen$=n$+",s,w"
1550 open1,pe,is,n$:poke3112+sc,c
1560 ifi1=0then1580
1570 fori=0to(i1*40)-1:da=peek(in+i):gosub1600:next
1580 fori=0to959:da=peek(3112+i):gosub1600:next
1590 fori=0to(i2*40):da=peek((fi-i2*40)+i):gosub1600:next:goto1640
```

```
1600 ifda=31orda=32thenda=13:i=int((i/40)+1)*40-1:goto1630
1610 ifda>0andda<32thenda=da+64:goto1630
1620 ifda>64andda<97thenda=da+128
1630 a$=chr$(da):print#1,a$;:return
1640 close 1:color1,6,6
1650 gosub2060:goto220
1660 :
1670 rem ** load **
1680 :
1690 h=1:i=0:ifpe=1thenprintchr$(19)+chr$(19):poke2021,0:printchr$(19):is=0:goto
1740
1700 op$="Load":gosub1980
1710 iflen(n$)=0then1820
1720 is=5:open5,8,15,"i0:":close5
1730 ifds>0thengosub1310:goto1820:elsen$=n$+",s,r"
1740 open1,pe,is,n$:gosub1070
1750 dountilst=64or(i/40)>152
1760 get#1,a$:printchr$(19)+a$
1770 ifasc(a$)=13thenda=31:goto1830:elseda=peek(3112)
1780 ifda=32thenda=96
1790 pokein+i,da:i=i+1
1800 loop:poke3112+sc,c+128
1810 close 1:color1,6,6
1820 gosub2060:i1=int(i/40):goto220
1830 pokein+i,da:forzy=i+1toint((i/40)+1)*40-1:pokein+zy,32:next:i=zy:goto1800
1840 :
1850 rem ** cancella linea **
1860 :
1870 poke205,int(sc/40)+1:sys56992:gosub1160:sc=int(sc/40)*40
1880 c=peek(3112+sc):poke3112+sc,c+128:goto220
1890 :
1900 rem ** insert linea **
1910 :
1920 ifi1+i2<152thenpoke3112+sc,c:poke205,int(sc/40)+1:elsegosub1310:goto220
1930 gosub1090:sc=int(sc/40)*40:fori=0to39:poke3112+int(sc/40)*40+i,96:next
1940 c=96:poke3112+sc,224:goto220
1950 :
1960 rem ** nome file **
1970 :
1980 n$="":color1,6,6:char1,18,0,chr$(18)+op$+":                    "+chr$(146)
1990 getkeyb$:a=asc(b$)
2000 ifa=20thenb$="":ifh-1>0thenh=h-1:poke3094+h,160:n$=left$(n$,len(n$)-1):goto
1990
2010 ifa=13thenreturn
2020 ifinstr(le$,b$)=0thenb$="":goto1990
2030 ifh>16then1990
2040 char1,22+h,0,chr$(18)+b$:n$=n$+b$:h=h+1:goto1990
2050 :
2060 char1,0,0,chr$(18)+" Scriptor v1.1                           ":poke3111,160
2070 poke2087,101:color1,2,6:gosub1070:return
2080 :
2090 rem ** stampa **
2100 :
2110 open1,4,7:close1:ifst<>0thengosub1310:goto220
2120 open1,4,7
2130 ifi1=0then2150
2140 fori=0to(i1*40)-1:da=peek(in+i):gosub1600:next
2150 fori=0to959:da=peek(3112+i):gosub1600:next
2160 fori=0to(i2*40):da=peek((fi-i2*40)+i):gosub1600:next
2170 close 1:goto220
2180 end
```

IL CX, QUESTO SCONOSCIUTO

Un mini viaggio nell'aerodinamica per conoscere più da vicino un coefficiente fin troppo inflazionato

di **Alessandro Diano**

Non molto tempo fa mi trovai ad accompagnare un mio amico presso il concessionario dove aveva prenotato la sua ultima "fiamma" automobilistica, per sbrigare le ultime formalità burocratiche per la consegna della vettura.

Durante l'attesa, mentre mi divertivo a curiosare tra i cruscotti delle auto in esposizione, sentii uno dei venditori borbottare definizioni varie ad un cliente evidentemente interessato ad una delle quattroruote presenti.

Mi avvicinai per ascoltare la spiegazione e notai con piacere il vertere della stessa sull'aerodinamica del mezzo anche se, ben presto, mi appassionò maggiormente il volto del cliente che "appassiva" sempre di più ad ogni definizione del pur bravo venditore.

Termini come "resistenza di scia", "penetrazione aerodinamica" e simili dovevano evocare nel poveretto fantasie incredibili a giudicare dalle altrettanto incredibili mimiche facciali che lasciavano trasparire un "ma parla come mangi!" nei confronti del venditore che, da parte

SCHEDA TECNICA

Software applicativo per impieghi tecnici

Hardware richiesto: C/16, Plus/4, C/128, ma facilmente adattabile al C/64

Richiede (per il solo C/64) Tool grafici di adattamento: routine grafiche di Toma, GW-Basic, Simon's Basic o altri

Anche i programmi pubblicati in queste pagine sono contenuti nel disco "Directory" di questo mese.

sua, continuava imperterrito nell'inutile esposizione.

Chiesi poi al mio amico (esperto aeronautico) che cosa ne pensasse dei venditori d'auto e di quel signore in particolare; mi rispose che non ne capiva nulla di aerodinamica ma, avendo imparato qualche termine di consumo, si dimostrava irrefrenabile nello sciorinarli senza pietà ai clienti che gli capitavano sotto tiro.

"Ha allontanato più clienti lui spiegando alla gente cos'è il Cx che non le cronache quotidiane di incidenti mortali!", concluse il mio amico, facendomi sorridere non poco.

La teoria

Sfortunatamente coloro che in questo settore hanno la possibilità di distinguere i bestemmiatori dai possessori di causa cognitiva non sono molti; per contro i blasfemi prolificano spaventosamente, in particolare tra i venditori di automobili anche se, ovviamente, ci sono le doverose eccezioni.

Infatti, provando a chiedere al prossimo che cosa sia il Cx, si ottengono fantasiose definizioni che persino Wanna Marchi, nota esperta di aerodinamica, saprebbe dare in maniera più corretta.

Ad onor del vero la cosa non è chiara neppure ad alcuni ingegneri aeronautici, perchè il fratello gemello aeronautico del Cx è noto con il nome di coefficiente di resistenza (abbreviato in Cr) oppure, (che è lo stesso), coefficiente di drag (Cd) dall'equivalente inglese del termine italiano "resistenza".

Si tratta di un numero adimensionale (ovvero senza unità di misura) ottenibile dalla somma dei coefficienti di resistenza parassita o passiva (in pratica la forma del corpo sia esso aereo, sasso, ferro da stiro eccetera) ed indotta, dipendente quest'ultima dall'assetto di volo del

corpo (il Cp aeronautico detto coefficiente di portanza) oltre che da uno specifico termine detto "allungamento alare" il quale, immaginando l'ala di un velivolo come un rettangolo, indica quante volte il lato minore (corda alare) è contenuto nel maggiore (apertura alare); esteticamente si traduce in un'indicazione della "snellezza" di un'ala.

Chiaramente in un'automobile non è presente nessuna ala, benchè la resistenza esiste, eccome, con tutti i coefficienti del caso; ecco allora che si ricava il Cx, Cr, Cd dalla ben nota (per gli aeronautici) relazione che a scuola si recita come:

"Resistenzaugualeromezziviquadro essecierre!"

che scatena inevitabilmente l'applauso dei presenti se la recita termina senza papere.

Ricavando il valore del Cr, che ormai è possibile assimilare al Cx, si ottiene la relazione:

Cr = (2*R) / (ro* V↑ 2* S)

nella quale:

R= Resistenza aerodinamica
ro= Densità del fluido nel quale ci si trova (generalmente aria)
V= Velocità del corpo
S= Superficie di riferimento

Quest'ultima (cioè S), è una delle "trappole" delle quali si servono le case automobilistiche per il calcolo del Cx; in aeronautica, infatti, si specifica SEMPRE la superficie rispetto alla quale si è calcolato il Cr (generalmente è quella alare), in quanto, come ben si nota, considerando una superficie molto grande il Cx cala sensibilmente, così come con densità e/o velocità maggiori.

Solitamente si impiega la superficie frontale dell'automobile anche se esistono Cx calcolati rispetto alla suddetta più la superficie del cofano anteriore (orrore!) oppure valori non riferiti alla configurazione commerciale della vettura.

L'idea

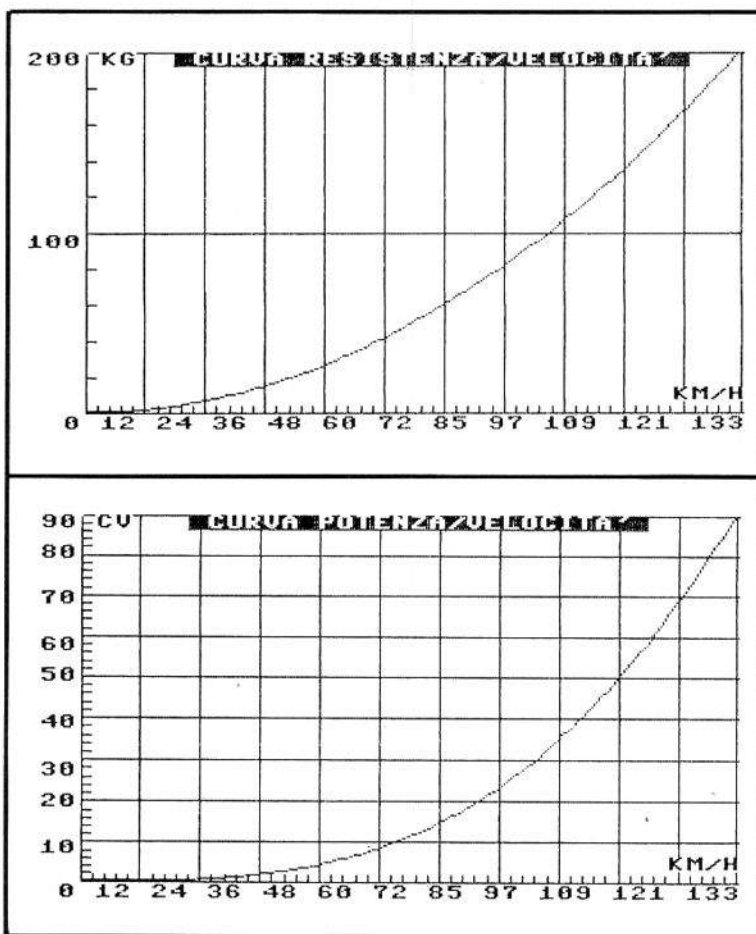
Sarebbe bello, pensavo, se, così come si può calcolare la convenienza tra diesel e benzina, le curve di consumo e così via, si potesse avere una qualche possibilità di analizzare (molto semplicemente) le prestazioni aerodinamiche di aerei, auto, biciclette, motorini, nasi adunchi eccetera.

E' bastato un esercizio proposto sul lavoro a farmi intravedere la possibilità di stendere un programma che avesse la possibilità di risolvere questo e tutti gli altri quesiti del genere, visto che si trattava degli stessi calcoli, pur se con dati differenti: il tipico problema da computer.

Il programma

Il programma proposto permette quindi di calcolare, con una buona approssimazione, il Cx di qualunque corpo: da quello umano agli aerei, automobili, motociclette e vecchie nonne.

In linea di massima è perfino possibile trovare il Cx di un sottomarino in acqua o di un'ape nel miele, a condizione di conoscere le densità, rispettivamente, di acqua e miele.

Questa funzione, sostanzialmente molto semplice, viene svolta da entrambi i programmi presentati; solo il secondo, però, permette il tracciamento di due grafici piuttosto utili per chi vuole spaventarsi su come,

alle alte velocità, aumentano terribilmente la resistenza e la potenza impiegate.

I possessori di Commodore 128 e di Plus/4 potranno terrorizzarsi tranquillamente osservando come raddoppia la resistenza all'aumentare della velocità di poche decine di Km. orari, mentre i C/16-isti dovranno fare qualche miracolo per liofilizzare il programma nei 2045 bytes che l'alta risoluzione lascia liberi; per chi ha il solo C/64 sarà comunque possibile la conversione del listato con l'ausilio dei vari tools grafici presenti in commercio.

L'utilizzo

Per semplificare al massimo gli input dell'utente vengono mantenuti, per i vari parametri, le unità di misura più comuni: verrà chiesta la velocità massima del corpo in Km/h, la potenza (sempre massima, ovviamente) in cavalli (CV) e la superficie di riferimento (supposta rettangolare) sotto forma di larghezza ed altezza; valori che il programma provvederà poi ad elaborare correttamente.

La faccenda, ovviamente, è approssimata visto che ben difficilmente la sezione frontale di un mezzo sarà perfettamente rettangolare anche se l'approssimazione è alquanto accettabile.

I più precisi potranno sempre inserire fattori correttivi o, meglio ancora, determinare il tradizionale integrale per il calcolo esatto dell'area in questione e rispondere a "Larghezza?" ed "Altezza?", rispettivamente con il valore trovato ed il numero uno, in modo da avere l'esatta superficie di riferimento per il calcolo del Cx.

Una volta mostrato il valore del Cx, il programma presenterà un menù di scelta comprendente il grafico di variazione della resistenza aerodinamica in funzione della velocità, l'analogo potenza/velocità e le opzioni di restart e di fine lavoro.

Scegliendo uno dei due grafici lo schermo verrà disabilitato per velocizzare le procedure di calcolo la cui durata sarà direttamente proporzionale alla velocità massima introdotta: calcolare il grafico dello Space Shuttle che ha velocità orbitali di Mach 22 (circa 27.000 Km/h) richiederà un tempo spaventosamente superiore a quello dell'utilitaria da 140 Km/h. (attenzione, comunque, alla densità dell'aria riscontrabile alle varie altezze).

Infine, alla "ricomparsa" dello schermo, il computer aspetterà la pressione di un tasto qualsiasi per lasciare la pagina grafica e ritornare al precedente menù.

I grafici di queste pagine si riferiscono a due autovetture di caratteristiche diametralmente opposte: la Ferrari Testarossa (Cx=0.396) e la quasi-fuoristrada Mitsubishi Pajero Wagon (Cx=0.623).

I valori del Cx, come spiegato in precednza, si discostano da quelli dichiarati dalle case costruttrici, a causa dell'approssimazione introdotta.

Qualche "dritta"

Coloro i quali posseggono un Commodore Plus/4 (o C/16) devono rammentarsi, prima del RUN, di eliminare la linea 170 come, del resto, chiaramente indicato nella REM.

Il programma è volutamente diviso in blocchi per poter facilitare al massimo la possibilità di "metterci le mani" da parte di chiunque; tra le modifiche principali suggerirei l'aggiunta di una routine di stampa della pagina in alta risoluzione per poterla consultare in seguito, magari quando sorge la voglia di schiacciare un attimino...

Se la memoria non è un problema, sarebbe anche interessante mettere

```
100 REM CALCOLO DEL CX
110 REM (C) 1988 A. DIANO - E. DENTI
120 REM PER QUALSIASI COMPUTER
140 HP=75: REM KG*M/SEC IN OGNI CV (ST)
150 RO=1/8: REM DENSITA' ARIA A 0 MT.
170 REM INGRESSO PARAMETRI
190 PRINT CHR$(147)
200 PRINT TAB(9);"CALCOLO DEL CX"
220 INPUT "VELOCITA' MASSIMA (KM/H)";VM
240 INPUT "POTENZA (CV)";PW
260 INPUT "LARGHEZZA (MT)";LR
280 INPUT "ALTEZZA (MT)";AL
310 REM COMPUTAZIONE
330 K=2*HP*(3.6↑3)/RO: REM COSTANTE
340 CX=K*PW/((VM↑3)*LR*AL)
360 REM FORMATTAZIONE OUTPUT
380 A$="":IF INT(CX)=0 THEN A$="0"
390 A$=A$+MID$(STR$(CX),2,4)
410 REM OUTPUT
430 PRINT "IL CX E' ";A$,"("CX")"
460 REM FINE
480 PRINT "ANCORA (S/N)"
490 GET A$: IF A$="S" THEN RUN
500 IF A$<>"N" THEN 490
510 PRINT: END
```

una doppia istruzione DIM in linea 290 ed una variabile con funzione di flag che riveli se il vettore del grafico richiesto è già stato riempito, saltando in caso positivo direttamente alla routine di tracciamento per risparmiare un secondo tempo d'attesa per gli stessi calcoli.

Se invece il calcolatore posseduto è un C/16 inespanso sarà opportuno eliminare REMarks, spazi e linee superflue, condensando le rimanenti in più istruzioni per linea.

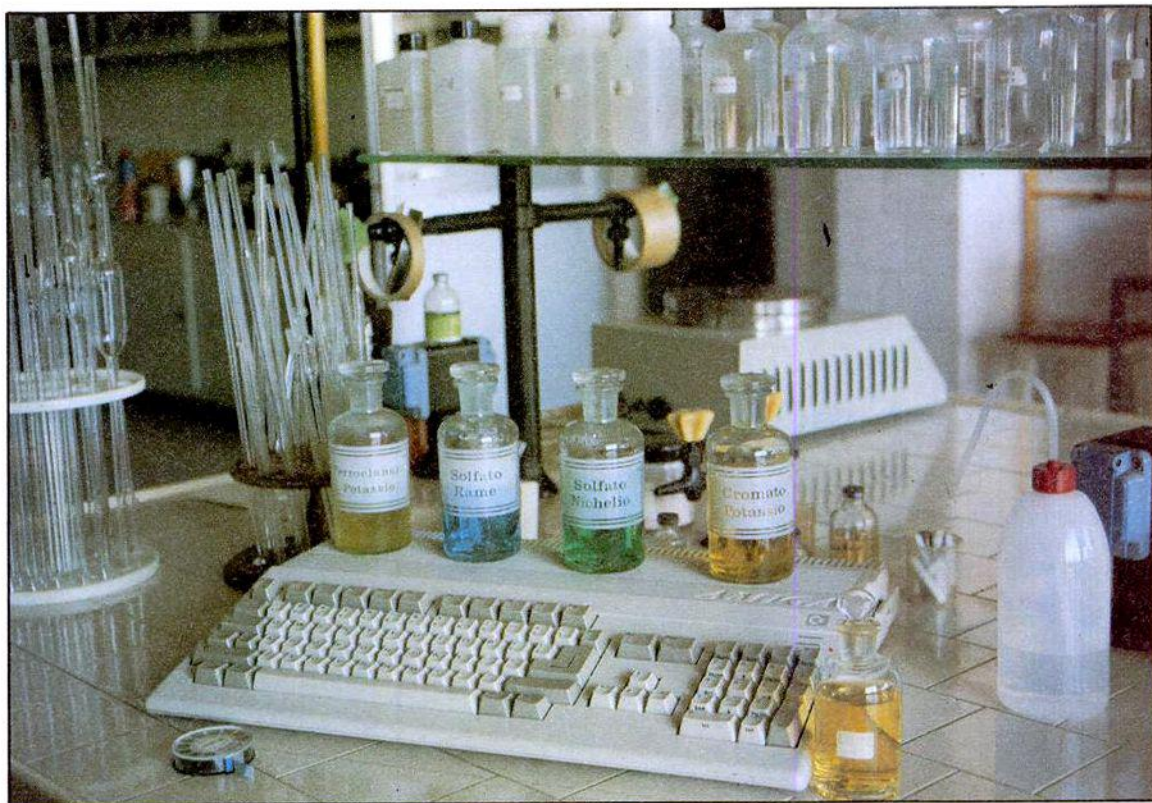
E' inoltre consigliabile commutare le routine di riempimento del vettore (510-530 e 660-670) con un plottaggio direttamente in alta risoluzione (780-800) per limitare l'impiego della memoria, altrimenti insufficiente.

Credits

Le relazioni matematiche impiegate nel programma sono sostanzialmente tre: alle righe 330-340, 530 e 670 per il calcolo rispettivo di Cx, resistenza e potenza; sebbene presenti (in altra forma...) su testi aeronautici e non, sono state elaborate ed adattate per lo scopo specifico di calcolare dei valori chiedendo all'utente dati facilmente reperibili (unità di misura comprese!).

Il lavoro di "adattamento" mi è stato possibile grazie soprattutto al preziosissimo aiuto dell'Ingegnere Eugenio Denti della Normale di Pisa che colgo l'occasione per ringraziare ancora una volta; basti sapere che tali relazioni calcolano un Cx dichiarato di 0.31 come un buon 0,35: 1/25 di errore è persino minore della tolleranza dei calibri di misurazione a nonio ventesimale impiegati in molte discipline tecniche...

```
100 REM CALCOLO CX & CURVE R/V E P/V
110 REM PROGRAMMA : PER.AER. A.DIANO
120 REM CONSULENZA: ING.AER. E.DENTI
130 REM PER C=128 - PLUS/4 - (C=16)
140 :
150 CLR
160 FS=65286: REM SOLO PER PLUS/4 & C=16
170 FS=53265: REM SOLO PER C=128
180 COLOR 4,1: COLOR 0,1: COLOR 1,2
190 HP=75:REM KG*M/SEC IN OGNI CV (SISTEMA TECNICO)
200 RO=0.125: REM DENSITA' ARIA A QUOTA ZERO
220 REM INGRESSO PARAMETRI
240 SCNCLR: PRINT TAB(9);"-=* CALCOLO DEL CX *=-"
250 PRINT: PRINT: INPUT "VELOCITA' MASSIMA (KM/H)";VM
260 PRINT: INPUT "POTENZA (CV)";PW
270 PRINT: INPUT "LARGHEZZA (MT)";LR
280 PRINT: INPUT "ALTEZZA (MT)";AL
290 PRINT: PRINT: DIM P(VM)
310 REM CALCOLO DEL CX
330 K=2*HP*(3.6↑3)/RO: REM COSTANTE
340 CX=K*PW/((VM↑3)*LR*AL): PUDEF " .,"
350 PRINT USING "IL CX VALE ###.###";CX
370 REM OPZIONI
390 PRINT: PRINT: PRINT "1] GRAFICO RESISTENZA/VELOCITA'"
400 PRINT: PRINT "2] GRAFICO POTENZA/VELOCITA'"
410 PRINT: PRINT "3] ALTRA ELABORAZIONE"
420 PRINT: PRINT: PRINT "0] FINE LAVORO"
430 GRAPHIC 0
440 GET A$: IF A$<"0" OR A$>"3" THEN 440
450 W=VAL(A$): ON W GOSUB 500,650,150
460 IF W THEN 430: ELSE GRAPHIC CLR: SCNCLR: END
480 REM CURVA RESISTENZA/VELOCITA'
500 GOSUB 820: REM FAST
510 K=HP*3.6: REM COSTANTE
520 FOR J=0 TO VM
```

```
530 P(J)=K*PW*J*J/(VM↑3): NEXT J
540 GOSUB 870: REM CALCOLO DEL GRAFICO
560 REM ETICHETTE
580 CHAR,5,0,"KG": REM VERTICALE
590 CHAR,36,22,"KM/H": REM ORIZZONTALE
600 CHAR,9,0," CURVA RESISTENZA/VELOCITA' ",1: REM TITOLO
610 GOTO 780: REM DISEGNA IL GRAFICO
630 REM CURVA POTENZA/VELOCITA'
650 GOSUB 820: REM FAST
660 FOR J=0 TO VM
670 P(J)=PW*(J↑3)/(VM↑3): NEXT J
680 GOSUB 870: REM CALCOLO DEL GRAFICO
700 REM ETICHETTE
720 CHAR,5,0,"CV": REM VERTICALE
730 CHAR,36,22,"KM/H": REM ORIZZONTALE
740 CHAR,10,0," CURVA POTENZA/VELOCITA' ",1: REM TITOLO
760 REM DISEGNA IL GRAFICO
780 LOCATE 34,190: FOR X=0 TO VM
790 DRAW 1 TO X/UX+34,190-(P(X)/UY)
800 NEXT X: POKE FS,27: REM SLOW
810 GETKEY A$: RETURN
820 POKE FS,11: REM FAST
830 RETURN
850 REM ROUTINE DI CALCOLO
870 MX=P(0): MN=P(0): FOR K=0 TO VM
880 IF P(K)>MX THEN MX=P(K)
890 IF P(K)<MN THEN MN=P(K)
900 NEXT K
920 REM SCELTA NUMERO SUDDIVISIONI
940 A=P(VM): D=0
950 IF (INT(A))<10 THEN 970
960 A=A/10: D=D+1: GOTO 950
970 A=INT(A+1): NS=A*5
990 REM UNITA' ORIZZONTALI & VERTICALI
1010 UY=(MX-MN)/190: UX=VM/285: GRAPHIC 1,1
1030 REM TACCHE VERTICALI
1050 NP=190/NS: FOR K=0 TO NS
1060 DRAW 1,35,190-K*NP TO 38-284*(K/5=INT(K/5)),190-K*NP: NEXT K
1080 REM CARATTERI PER OGNI SUDDIVISIONE
1100 Q=NP*5/8: FOR K=A TO 0 STEP -1
1110 A$=MID$(STR$(K*10↑D),2,4)
1120 CHAR,4-LEN(A$),24-Q*K,A$: NEXT K
1130 NS=15: REM SUDDIVISIONI ORIZZONTALI
1140 J=VM/NS:A$=""
1160 REM SCALA DELLE VELOCITA'
1180 FOR K=1 TO NS: A$=A$+STR$(INT(J*K)): NEXT K
1190 L=LEN(A$): IF L>36 THEN NS=NS-1: GOTO 1140
1200 NS=NS*5: NP=285/NS: FOR K=0 TO NS
1210 DRAW 1,34+K*NP,190 TO 34+K*NP,186+185*(K/5=INT(K/5)): NEXT K
1220 CHAR,40-LEN(A$),24,A$: RETURN
```

COME FUNZIONA IL "C"

Chi conosce solo il Basic ha una conoscenza incompleta del concetto di "funzione", frequente nel nostro amato "C" ed in altri linguaggi come il Pascal o il Modula 2

di **Luigi Callegari**

Dopo avere visto che cosa sono i compilatori (C.C.C. n.50), quali sono i tipi di variabili ed i loro operatori generici che di norma utilizza il linguaggio "C" (C.C.C. n.51), manca "solo" di spiegare il concetto di funzione, per avere a disposizione tutti i mattoni necessari per incominciare a scrivere qualche cosa di significativo e, magari, utile...

Istruzioni e funzioni

Le istruzioni, o "parole-chiave" accettate dallo standard C, sono circa una trentina. Molto poche, dunque, perchè mancano, ad esempio, le istruzioni per gestire le stringhe, così come non esiste una specifica istruzione per l'elevamento a potenza (non è disponibile nemmeno l'operatore specifico). Anzi, persino la "printf", incaricata di stampare un output, simile alla PRINT del vecchio Basic, non è una parola-chiave riservata al sistema.

Il linguaggio C ottiene, però, tutte queste cose, e molte altre, tramite "funzioni", appunto, che sono le sorelle evolute dei sottoprogrammi (o subroutines) dei Basic e Fortran (ma chi conosce Pascal o AmigaBasic ne ha già una chiara idea).

Chiariamo subito una convenzione: nei testi sul C una funzione solitamente si scrive seguita da una coppia di parentesi tonde. Quindi scriveremo...

void

...che è una parola-chiave riservata, e...

printf()

...che è una funzione.

Le funzioni possono essere di due tipi:

- *Di libreria*
- *Scritte dall'utente*

Ogni compilatore (ve ne sono diversi, proposti da altrettante software house) viene fornito con un certo numero di librerie standard, più altre librerie di supporto che consentono di sfruttare le caratteristiche hardware della macchina.

Il file di libreria "stdio.h", contenente definizioni e funzioni generali per lo I/O, viene fornito, ad esempio, con qualunque package, a causa della sua vitale importanza.

Insieme ai compilatori per Amiga, inoltre, viene fornita una libreria chiamata "Exec" che permette di usare alcune funzioni, specifiche per Amiga, che permettono di gestire direttamente una parte del sistema operativo, senza esser costretti a riscriverne una parte.

Per le macchine MS/DOS esistono anche librerie che permettono di crear programmi che si presentano in un ambiente "a finestre", come lo standard di Intuition-Amiga.

Si noti che tali funzioni e strutture di dati vengono descritte non nei manuali che accompagnano il programma, ma nella documentazione ufficiale della Commodore / Addison-Wesley. Ciò non avviene, invece, nelle librerie tipicamente fornite con sistemi come lo MS/DOS, dove le librerie di funzioni speciali sono fornite solitamente a parte, e con ponderosi manuali descrittivi.

C/64 e Linguaggio C

Scrivere programmi in C sul caro vecchio C/64 è una vera impresa, come stiamo sperimentando a nostre spese.

Esistono solo due (vecchie) versioni di questo linguaggio che, se non bastasse, sono proposte secondo regole assolutamente non-standard: una di esse limita persino i parametri di printf() e funziona solo con numeri interi; dulcis in fundo esse non sono neppure importate ufficialmente.

Per giunta, rivolgendoci come al solito al mercato parallelo (leggi: pirateria ed appassionati), mai ci è capitato di ricevere così tanti sguardi incuriositi e sospettosi dalle persone che posseggono migliaia di dischetti di programmi per C/64 da vendere a poche lire al quintale.

Alcuni non sanno nemmeno che cosa sia il C (e questo succede anche in alcuni negozi Commodore Point...), molti lo hanno cercato inutilmente tra le loro collezioni di floppy che occupano intere stanze: NESSUNO sembra, però, disporre della indispensabile documentazione originale (magari in fotocopia), per cui il panorama è tragico.

Da tener presente che compilare programmi in C con un C/64 o C/128 è una delle azioni che potrebbe farvi perdere gli anni più belli della vostra gioventù: nel tempo di una compilazione di un programma di lunghezza medio-breve si ha il tempo di leggersi la Divina Commedia (comprese le note a piè di pagina). La maggior parte del tempo necessario a scrivere un programma, una volta che si è miracolosamente capito come si compila, non è, come di consueto, occupata dalla correzione degli errori, ma dagli adattamenti necessari per via delle pesanti limitazioni rispetto allo standard Kernighan & Ritchie (a parte la tartarugosità dei floppy drive).

Ms-Dos e linguaggio C

Diverso, o meglio, opposto, il discorso per i possessori di macchine MS/DOS. C'è solo l'imbarazzo della scelta di programmi, anzi, dei packages completi di sviluppo in linguaggio C.

Molti programmi sono importati e distribuiti capillarmente da case italiane (la Borland, ad esempio), ad un prezzo certamente accettabile per programmi originali di questa qualità. Esistono almeno sette ottime versioni per MS/DOS del linguaggio, tutte standard "Kernighan & Ritchie" estese, funzionanti egregiamente anche con un semplice PC-compatibile da 640K dotato di un solo drive (ma i professionisti collegano sempre un hard-disk). Tra queste versioni esiste anche la Lattice, che è fondalmentalmente identica a quella per Amiga, perciò i discorsi generali che facciamo in queste pagine possono essere "riciclati" anche dagli utenti MS/DOS. Del resto C è un linguaggio piuttosto standardizzato, quindi finchè non si parla delle librerie fornite con uno specifico compilatore per una data macchina, i concetti e gli esempi sono sempre gli stessi.

La nostra prima funzione

Un programmino tipico, molto noto a chiunque inizi lo studio di "C", è il primo di queste pagine.

Questo programma, benchè "minimo", è sufficiente a dimostrare l'uso di una funzione della libreria standard, cioè printf(), simile alla PRINT del Basic.

Il programma fa leggere, al compilatore, il file di libreria chiamato "stdio.h" (che è l'abbreviazione di "standard I/O"), in modo da rintracciare le informazioni necessarie per definire e fare "funzionare" printf(); in seguito chiama questa funzione per stampare la semplice stringa "Ciao, bello!".

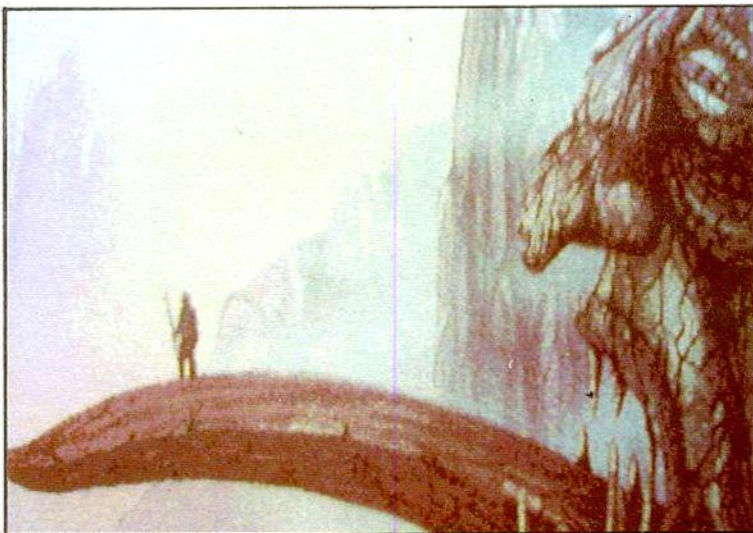
In realtà abbiamo attivato due funzioni: la seconda è main(). Per convenzione il nostro programma deve sempre definire la funzione principale. Infatti, quando il programma sarà stato compilato e trasformato in un codice eseguibile, lanciandolo verrà eseguita sempre per prima questa funzione, che tipicamente ne chiama altre (di libreria e scritte dall'utente) per raggiungere il suo scopo. Tale funzione può anche non essere la prima nel listato sorgente.

```
#include <stdio.h>
main()
{
printf("Ciao, bello! \");
}
```

Programma n.1

Vediamo ora un esempio di funzione scritta da noi, cioè l'elevamento a potenza di numeri interi (programma n.2).

```
pot(a, n)
int a, n;
{
int p;
for ( p=1; n>0; --n)
p = a * x;
return(p);
}
```

Programma n.2

La parolina "pot" indica il nome da noi assegnato alla funzione, cioè quello che useremo per "chiamarla" quando ci serve. Tra parentesi vanno inseriti i parametri necessari per arrivare alla funzione (in questo caso sono due) simboleggiati dalle lettere "a" e "n"; si noti anche che non è presente il carattere di punto e virgola (;). La linea successiva spiega come devono essere quei parametri: nel nostro caso sono due numeri interi. Aperta la parentesi graffa inizia il "corpo" della funzione, ovvero la sequenza di operazioni da svolgere quando la si chiama: nel nostro caso è un ciclo for (simile al FOR...NEXT del Basic).

La funzione si chiude con la parola return che comprende, tra parentesi, il parametro da restituire al programma chiamante, nel nostro caso il contenuto della variabile "p". Tale istruzione può anche essere assente, nel quale caso la funzione è detta di tipo "void", cioè che non restituisce alcun valore.

Prima del nome della funzione si dovrebbe indicare, appunto, qual'è il tipo (intero, virgola mobile, carattere...) del valore restituito alla funzione. Se si tralascia, il compilatore assume che la funzione elabora un numero intero.

Un programma che volesse fare uso della funzione pot, per calcolare il valore di due elevato alla terza e il contenuto della variabile "base" elevata alla quinta, può essere, ad esempio, il programma 3.

```
main()
{
int c,d,base;
c = pot(2,3);
d = pot(base,5);
}
```

Programma n.3

Questo programmino, che ovviamente deve comprendere anche la funzione pot posta prima o dopo main(), evidenzia come si passano i parametri alla funzione e come si sfruttano i valori elaborati. Le variabili "c" e "d" conterranno i risultati che arrivano da return(p). Per quanto riguarda i parametri passati, coloro i quali hanno già usato in Basic le funzioni definite dall'utente (DEF FN, FN) avranno già afferrato la similitudine: nella funzione i valori che arrivano dal programma chiamante (2,

Portatilità vò cercando

Ripetiamo: chi scrive programmi standard, senza ricorrere a funzioni di librerie specializzate, può usare Amiga con uno dei due compilatori C (Lattice o Manx) senza problemi, ispirandosi a qualunque libro presente sul mercato. Chi possiede, invece, una macchina MS/DOS, sistema ormai non giovane pur se collaudatissimo, dispone anche di parecchie dozzine di testi specialistici e dedicati (in inglese) oltre ad alcuni volumi tradotti di recente.

Chi dispone del C/64, deve, se riesce a trovarlo, accontentarsi di giochicchiare col compilatore dopo aver trafficato non poco ad imparare come si usa. Ad eccezione di questi ultimi, che dispongono di compilatori ridotti e non standard, tutti possono scrivere programmi che vanno bene per qualunque macchina: basta che rispettino le indicazioni date da Kernighan & Ritchie e leggano bene i manuali.

Precisiamo, comunque, che quando si incomincia a scrivere, per Amiga o per MS/DOS (C/64 non lo permette) sfruttando le librerie speciali che permettono di sfruttare il Sistema Operativo e l'hardware specifico, si incontrano regolarmente problemi di incompatibilità.

Parlando per esperienza, quasi nessun programma, appena evoluto, scritto per Amiga, che usi le librerie standard (descritte nella documentazione ufficiale) sotto uno dei due compilatori funziona anche compilandolo con l'altro! A buon intenditor...

3 e "base", 5) vengono assegnati alle variabili indicate nella prima linea di definizione. Si noti che è possibile passare valori di tipo diverso, quasi sempre, perchè il C effettua una conversione automatica dei tipi (casting, in inglese), come abbiamo già detto in passato.

Ciò significa che nella funzione main() chiamante la variabile "base" avrebbe anche potuto essere di tipo float, double, oppure, al limite, char. In ogni caso il compilatore avrebbe provveduto a convertire il valore in un numero intero, brontolando con un "WARNING" dell'operazione sospetta che è stato costretto a compiere, ma generando, tuttavia, un codice funzionante.

E' bene, ora, chiarire un concetto fondamentale del C: i valori vengono passati alle funzioni sempre per valore. Dunque, visto che alla funzione chiamata arrivano dei semplici numeri, essa non può modificare nulla nel programma principale. Solitamente, tutte le variabili usate nel corpo di una funzione sono "interne" ad essa e separate da quelle di tutte le altre funzioni, compresa main(). Ovviamente è possibile, con opportune specifiche, fare condividere alcune variabili da tutte le funzioni, ma è bene, allo stato attuale delle cose, soffermarsi sul concetto di "località" delle variabili all'interno di una funzione.

Ciò permette all'utente, come in AmigaBasic od in Pascal, di scrivere librerie di funzioni che possono essere inserite, e sfruttate, in più programmi, senza preoccuparsi molto delle variabili che usa, ma badando solo ai parametri necessari in input ed a quello (se è uno solo, come stiamo trattando per adesso) restituito in uscita.

Si noti che non è possibile restituire più di un valore, cioè indicare più di una variabile all'interno di return(). Ciò, ovviamente, non significa che una funzione non può trattare matrici o più parametri, ma solo che, in casi come questo, è richiesto l'uso dei puntatori di cui potremo parlare più avanti.

Naturalmente le funzioni possono "chiamarsi" tra di loro, visto che anche main() è, a tutti gli effetti, una funzione come le altre. Non è possibile, comunque, definire una funzione all'interno di un'altra; la ricorsione, cioè la chiamata, da parte di una funzione, di se stessa, è permessa entro certi limiti, determinati dalla quantità di memoria di "stack" accessibile dal programma al momento dell'esecuzione.

Ciò che manca, e manca tanto!, sulle funzioni, verrà affrontato gradatamente sui prossimi fascicoli di Commodore Computer Club.

Sviluppo incrociato

Vi è mai capitato di leggere il termine "cross-compiler" e di chiedervi che cosa significhi? Non significa che si tenta di attraversare la strada con un package di compilazione sotto il braccio, bensì che è possibile usare un computer per sviluppare programmi per una macchina di tipo diverso.

Ad esempio, il sistema operativo di Amiga è stato scritto su di un minicomputer SUN e poi trasferito su ROM. Parimenti moltissimi videogiochi del C/64 sono stati sviluppati dapprima su computer tipo MS-DOS o più grandi, con parecchi mega di memoria e veloci hard-disk, per consentire un lavoro estremamente più agevole.

Pare che stiano arrivando dei packages che permettono di sviluppare, su Amiga, programmi specifici per Ms-DOS (e viceversa), convertendone i formati di disco o trasferimento tramite RS-232.

UNA MEMORIA DI FERRO

Uno strumento utilissimo per creare programmi "permanenti"

di **Giancarlo Mariani**

Il mercato del C/64 è ormai invaso da cartucce varie, super-velocizzatori, iper-copiatori, sprotettori, e chi più ne ha più ne metta.

Sembra che le case produttrici di hardware facciano di tutto per produrre schede sempre più sofisticate e spesso, lasciatecelo dire, parzialmente... inutili, a causa del gran numero di funzioni presenti nella scheda rispetto a quelle che realmente servono spesso.

Questa volta parleremo di un accessorio che, anche se non super accessoriato come i precedenti, presenta ottime caratteristiche ed interessanti possibilità che lo pongono in primo piano rispetto ad altri accessori h/w.

L'oggetto misterioso di cui stiamo parlando è il programmatore di Eprom della MERLIN. Purtroppo la scheda risulta corredata di manuale scritto in tedesco, per cui, data la nostra ottima(!) conoscenza di questa lingua, ci scusiamo in anticipo con i lettori se non descriviamo tutte le caratteristiche dell'oggetto in questione che, tuttavia, è stato utilizzato con successo nel corso del test.

Come programmare una Eprom

Per programmare una Eprom, questa deve essere "vuota", ossia nuova oppure precedentemente cancellata.

Nel caso sia nuova, "dovrebbe" essere già vuota, benchè si possa

controllare tramite il programmatore stesso, come vedremo tra breve. Nel caso la Eprom sia invece già stata utilizzata almeno una volta, bisogna armarsi di lampada U.V. (ultravioletti) e di molta pazienza.

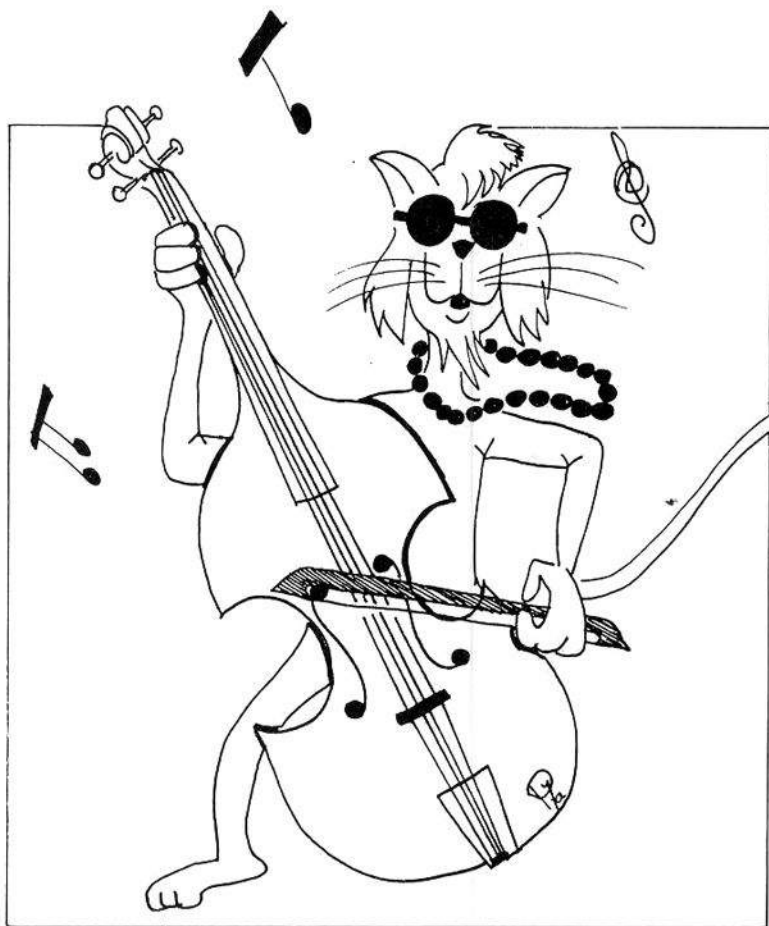
Per cancellare una Eprom è sufficiente posizionarla sotto la lampada, con la fessura in diretta esposizione alla luce UV, accendere la lampada e... aspettare. Il tempo necessario alla Eprom per "svuotarsi" dei propri contenuti è variabile, e dipende dal tipo di circuito integrato. Abbiamo provato, con successo, con un tempo prossimo ai 60 minuti; riteniamo, tuttavia, che questo sia la base da prendere in considerazione, aggiungendo o togliendo qualche decina di minuti a seconda della lampada usata. Le specifiche lampade cancellatrici per Eprom riportano i tempi di attesa che, ovviamente, sono minori di quelli da noi usati avendo a disposizione una semplice lampada abbronzante.

Una volta cancellata, la Eprom può essere inserita nel programmatore che, dimenticavamo, è composto da due congegni distinti. Il primo è una cartuccia, da inserire nel retro del C/64, che contiene il software necessario per far funzionare il programmatore.

Il secondo è una scheda, che costituisce il vero e proprio circuito programmatore, e che va inserita nella porta utente. Questa scheda è composta da vari integrati e componenti, che eseguono le varie temporizzazioni e conversioni di tensioni necessarie, e da uno zoccolo per integrati a 28 piedini munito di levetta per il bloccaggio / sbloccaggio della Eprom.

E' quindi possibile inserire anche le Eprom di capacità più bassa, dotate di soli 24 piedini.

Una volta inserita la Eprom, si può passare all'accensione del computer: non bisogna MAI, infatti, inserire o togliere una Eprom se il computer è acceso, dato che questa operazione potrebbe portare ad un danneggiamento della Eprom o, peggio, del programmatore.

La cartuccia, all'accensione, visualizza un menu (naturalmente scritto in tedesco) con tre opzioni: con la prima è possibile passare al menu di scelta del tipo di Eprom, con la seconda si accede al Basic del C/64 (con in più alcuni comandi Basic aggiuntivi), e con la terza si entra nel sistema normale C/64 + Basic, come se la cartuccia non fosse inserita.

La prima opzione presenta un menu completo di tutti i tipi di Eprom utilizzabili; bisogna riconoscere che la scelta è veramente molto vasta, poichè spazia dalle vecchie 2516 (2 kbytes) alle capienti 27512 (64 kbytes), con inclusi anche i tipi CMOS (27Cxx) le 28xx, le 687xx ed altri tipi di cui, lo confessiamo, non abbiamo mai sentito parlare.

In basso viene visualizzato il tipo di Eprom prescelto e la tensione di programmazione, che può essere di 5, 12.5, 21 e 25 volt. Bisogna fare particolare attenzione a questo parametro perchè una tensione troppo alta potrebbe danneggiare la Eprom, mentre una troppo bassa non riuscirebbe a programmarla. Se non siete sicuri della tensione di programmazione della Eprom in vostro possesso, potete iniziare da quella più bassa, per poi passare a tensioni più alte fino a quando la programmazione risulti corretta.

Una volta scelto il tipo di Eprom, appare una schermata con varie informazioni, oltre ad un menu. Viene visualizzato il tipo di Eprom selezionato, la sua capacità, i vari indirizzi di partenza e fine memoria e l'algoritmo da scegliere per la programmazione. Si hanno a disposizione ben 32k di memoria RAM del C/64 per scrivere i propri programmi: sono più che sufficienti perchè, in pràtica, per il C/64 non esistono cartucce che occupano più di 16k.

Le funzioni selezionabili sono le seguenti:

R: Lettura della Eprom.
P: Programmazione della Eprom.
L: Lettura, da disco, di un programma da trasferire su Eprom.
S: Scrittura da memoria a disco.
A: Cambio dei vari indirizzi e dell'algoritmo di programmazione.
T: Per cambiare il tipo di Eprom prescelto.
D: Directory.
K: Invia comandi al disco.
B: Controlla se la Eprom è vuota.
V: Verifica i dati scritti nella Eprom con quelli presenti in memoria.
G: Generatore di moduli (vedi dopo).
M: Monitor di l.m. (tipo Zoom).
X: Torna al menu iniziale.

Ammettendo di voler programmare la Eprom, e supponendo di averla già inserita nella scheda, la prima cosa da fare è utilizzare l'opzione B, che controlla se in ogni sua cella è presente il valore esadecimale FF. Fatto questo, ammettendo che non vi siano errori, si può passare alla programmazione vera e propria, tramite l'opzione P.

Il computer comincia, quindi, a trascrivere i dati dalla sua memoria alla Eprom, e l'operazione di trasferimento è confermata anche dall'accensione di un led giallo presente sulla scheda. Se alla fine della programmazione va tutto bene, vedremo il led spegnersi ed il C/64 visualizzare un rassicurante "KEINE FEHLER" (nessun errore); se qualcosa non va per il verso giusto il led inizierà a lampeggiare ed apparirà il messaggio "NICHT LEER" con l'indirizzo in cui si è verificato l'errore.

In questo caso vuol dire che la Eprom non era vuota, oppure era danneggiata, oppure avete scelto un tipo di Eprom diverso da quello inserito.

Il tempo di programmazione è variabile e dipende dalla Eprom usata; si aggira intorno ai 7 minuti per ogni 8k programmati. Questo tempo si può ridurre utilizzando uno dei tre algoritmi intelligenti di programmazione presenti nel software. Il manuale ne consiglia l'uso a seconda delle Eprom che, a volte, hanno tem-

Eprom

Tale sigla proviene dall'inglese "Erasable Programmable Read Only Memory", ossia "Memoria a sola lettura cancellabile e programmabile". Le ultime tre lettere della sigla (ROM) sono usate per specificare un certo tipo di memoria presente in tutti i computer, ossia la memoria a sola lettura, nella quale generalmente è contenuto il Sistema Operativo della macchina e, se presente, un interprete Basic.

Le ROM, a differenza delle loro "cugine" RAM (Random Access Memory= memoria ad accesso diretto, usate per contenere programmi e dati) hanno il vantaggio di non perdere il loro contenuto in mancanza di alimentazione elettrica, per cui un'informazione scritta in una ROM rimane per sempre. Ogni volta che accendiamo il C/64, ci troviamo in "ambiente" Basic, senza aver avuto la necessità di caricare l'interprete da disco o cassetta.

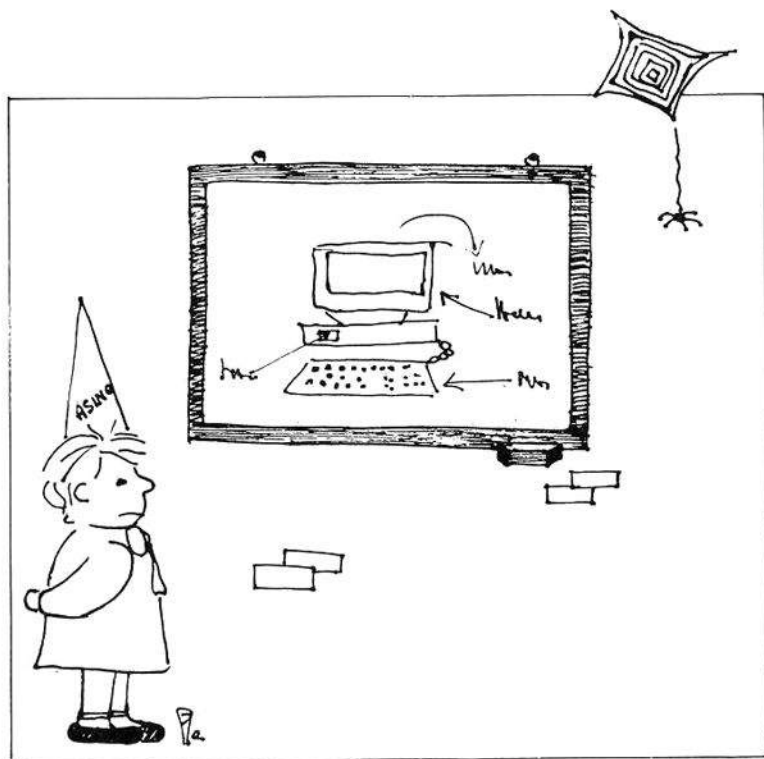
Lo svantaggio delle ROM è che, una volta programmate, non possono più essere modificate, ossia non c'è alcuna possibilità di variare il programma ivi contenuto. Se, infatti, per una qualsiasi ragione una ditta decidesse di commercializzare la versione aggiornata di un sistema operativo di un computer "basato" su una particolare ROM, dovrebbe contattare centri specializzati che, con complicati macchinari, riuscirebbero a programmare gli integrati, a patto però che siano vergini, ossia mai programmati prima, dato che la programmazione di una ROM brucia in modo irreversibile alcuni circuiti interni, in modo da ottenere in uscita i dati desiderati.

A mano a mano che i costi dei computer e delle periferiche diminuivano, anche l'utente medio di piccoli elaboratori si è trovato spesso nella necessità di scrivere i propri programmi su memorie di tipo ROM, magari per realizzare piccoli apparecchi a microprocessore dedicati al controllo di antifurti, di piccoli automatismi casalinghi, e così via. In questi casi non ci si può rivolgere alle ditte programmatrici di ROM, a causa degli alti costi dell'operazione per lavori su piccola scala; senza contare che, una volta ottenuto il chip, non è possibile apportare alcuna modifica al programma.

Per queste ragioni sono stati realizzati particolari dispositivi di memoria che uniscono, alla caratteristica di non perdere i dati quando viene a mancare l'alimentazione, l'interessantissima possibilità di essere riprogrammati a piacere mediante sempici apparecchiature elettroniche alla portata di tutti. Le memorie si possono suddividere in tre tipi principali:

1 - PROM (Programmable ROM)= memoria a sola lettura, programmabile. Questo tipo di memoria può essere programmato UNA VOLTA SOLA, usando un congegno elettronico. Volendo apportare modifiche al programma è necessario riscriverlo per intero su una nuova Prom.

2 - EAROM (Electrically Alterable ROM)= Memoria a sola lettura, alterabile elettricamente. Questa memoria può essere programmata più volte tramite segnali elettrici particolari, ma dato che non sono facilmente riproducibili, questo tipo di memoria non è adatto all'hobbysta.

3 - EPROM. Questa memoria è facilmente cancellabile e programmabile da chiunque, e l'hardware necessario si limita alla scheda descritta in questo articolo e ad una comune lampada a raggi ultravioletti.

pi di accesso minori. L'algoritmo di programmazione standard, invece, va bene per tutte le Eprom, ed è, in un certo senso, "intelligente" perchè prima di scrivere un dato esegue un controllo: se la cella contiene già il dato che si vorrebbe scrivere, la scrittura di quel dato viene evitata. Supponendo che la Eprom sia vuota, ossia contenga FF, il programmatore eviterà di inviare alla Eprom i bytes posti a FF.

Una volta programmata, la Eprom può essere verificata tramite l'opzione "V", che visualizza i messaggi descritti a seconda del risultato della verifica.

L'opzione R legge i contenuti della Eprom e li trasferisce in memoria. E' importante ricordare che tutte le operazioni descritte (letture, scritture, ecc.) vengono effettuate solo tra gli indirizzi specificati (BRENNSTART= indirizzo di partenza, BRENNENDE= indirizzo di fine), che comunque possono essere variati tramite l'opzione A.

Il monitor di l.m. è abbastanza simile al conosciutissimo Zoom, e quindi non ne descriviamo le potenzialità; ci limitiamo a menzionare alcune funzioni extra, non presenti su altri monitor. E' infatti possibile la lettura diretta (senza variare la locazione 1) della RAM sotto l'interprete Basic ed il S.O. del C/64, le varie conversione esadecimali / decimali, la visualizzazione e scrittura della memoria in notazione binaria e, dulcis in fundo, la possibilità di creare e correggere sprites.

Un'opzione interessantissima del programmatore è la "G" (generatore di moduli) non provata a fondo nel nostro test per mancanza di tempo. Con il generatore di moduli è possibile scrivere nella Eprom un "menu" da visualizzare non appena si accende il C/64, e che consente la scelta tra 8 programmi diversi. In pratica, noi potremo scrivere sulla nostra Eprom fino a 8 programmi (in Basic o in l.m.) e, tramite l'opzione suddetta, fare in modo che venga presentato, all'accensione, un menu tramite il quale far partire automaticamente uno dei programmi memorizzati.

Pensate a come potrebbe essere utile una simile caratteristica: ad esempio, potremmo mettere su di una Eprom di elevata capacità un monitor di l.m, uno sprotettore, un Word processor, un Basic esteso, ed averli sempre a disposizione ogni volta che accendiamo il computer. Questa è senza dubbio una delle caratteristiche più interessanti del programmatore che, disponendo perfino di un Freeze Frame incorporato (che consente la sprotezione di alcuni programmi) amplia le capacità della scheda.

Ma non è ancora finita!

All'accensione del computer, tramite l'opzione 2, è possibile accedere ad un Basic esteso. Con i nuovi comandi disponibili è possibile, da Basic, leggere la Eprom, programmarla, caricare programmi in l.m. presenti su di essa e, udite udite, perfino registrarvi DIRETTAMENTE programmi in Basic.

Il software provvede, infatti, a modificare i comandi LOAD e SAVE in modo da leggere e scrivere sulla Eprom come se fosse un disco. Tra i nuovi comandi è presente anche un comando per visualizzare la directory dei programmi presenti sulla Eprom. Non mancano, naturalmente, tutti i comandi per la gestione dei files, quali OPEN, GET#, PRINT#, INPUT#, CMD, CLOSE.

Naturalmente lo svantaggio è che i programmi ed i files, una volta scritti sulla Eprom, non si possono cancellare, a meno di non usare la tecnica descritta in precedenza.

Il manuale, a parte la lingua, è veramente completo, e riporta anche la piedinatura di tutte le Eprom utilizzabili.

Per maggiori informazioni:

Newel s.r.l.
Via Mac Mahon, 75
20155 Milano
Tel. 02/32.34.92
02/32.70.226

ESORDIENTI

- Il mezzo grafico

CAMPIONI

- Movie maker, secondo tempo
- Oltre le sbarre del video

ESPERTI

- Tutti i numeri del vostro Commodore 64
- Se la memoria non vacilla...
- Invito alla kermesse del kernel di Amiga

La Grande Libreria Systems

Autori Vari

64 Programmi per Commodore 64

Giochi, grafica, gestione delle stringhe, musica, numeri, gestionali.

Lire 4.800

Autori Vari

I miei amici C16 & Plus4

Un manuale pratico per padroneggiare il basic di questi computer.

Lire 7.000

Autori Vari

Strategie vincenti per Commodore 64

Le strategie per tutti i classici del videogioco: per giocarli, vincerli o programmarli.

Lire 5.800

Autori Vari

62 Programmi per il Vic 20, C16 e Plus 4

Giochi, grafica e routine per imparare a programmare.

6.500

Roberto Didoni, Guido Grassi

Utilities e giochi didattici

Raccolta di programmi pratici per tutti i Commodore e lo Spectrum.

Lire 6.500

Giovanni Mellina

Tutti i segreti dello Spectrum

4 passi nella Rom: come usare le più importanti routine del sistema operativo.

Lire 7.000

Roberto Didoni, Guido Grassi

Simulazioni e test per la didattica

Teoria e listati per Vic 20, C16, C64 C128 e Spectrum Sinclair.

Lire 7.000

Paolo Goglio

Impara giocando il basic dello Spectrum

Esercizi pratici per entrare nel vivo della programmazione.

Lire 7.000

Clizio Merli

µPascal per Commodore 64/128

Un manuale completo per il programma compilatore

Lire 7.000

Umberto Colapicchioni e Luca Galuzzi

Dal registratore al drive del C64

Tutti i segreti delle memorie di massa del Commodore 64

Lire 7.000

Autori Vari

ADA

Il linguaggio passepartout dei computer degli anni '80.

Lire 5.000

Clizio Merli

Il linguaggio PASCAL

Un manuale tascabile per lo studio e la programmazione.

Lire 5.000

Sì, voglio arricchire la mia biblioteca con i seguenti volumi al prezzo di copertina + lire 3.000 per spese di spedizione.

- ☐ **64 Programmi per Commodore 64**
- ☐ **Strategie vincenti per i tuoi videogames**
- ☐ **62 Programmi per Vic 20 C16 e Plus77**
- ☐ **Utilities e giochi didattici**
- ☐ **Tutti i segreti dello Spectrum**
- ☐ **Simulazioni e test per la didattica**
- ☐ **Imparare giocando il basic dello Spectrum**
- ☐ **I miei amici C16 e Plus4**
- ☐ **Pascal per Commodore 128**
- ☐ **Dal registrattore al drive del C64**
- ☐ **ADA**
- ☐ **Il linguaggio Pascal**

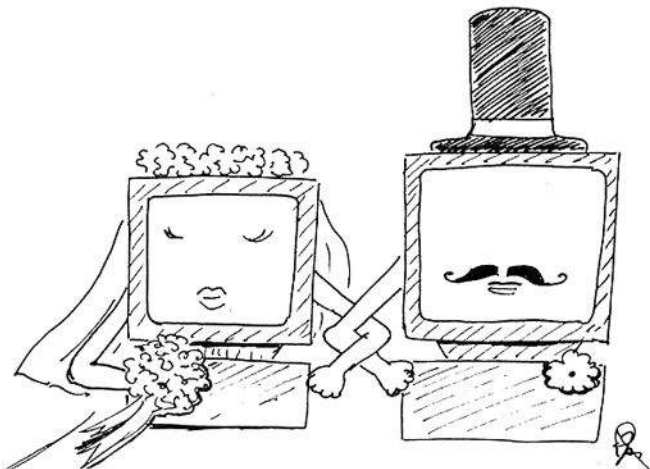
Nome ....

via .... **N.ro.** .... **telefono** ....

CAP .... **Città** ....

Su tale importo mi praticherete lo sconto del 10% in quanto abbonato a ☐ *Commodore Computer Club* ☐ *Personal Computer* ☐ *Com puter* ☐ *VR Videoregistrare.* Pertanto vi invio la somma soltanto di lire ....

Valore dell'ordine lire....................

Ritagliare e spedire in busta chiusa regolarmente affrancata a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

IL MEZZO GRAFICO

Far apparire sul video i soli caratteri alfanumerici è limitativo; ma, ricorrendo ai caratteri semigrafici...

Quando si desidera visualizzare i simboli che appaiono sulla tastiera, il metodo più semplice sembra essere quello di usare il comando Print associato ai vari simboli desiderati. Ad esempio:

Print "A" ; "n" ; "P"

...in cui, ovviamente, al posto dei caratteri "A", "n" e "P" vanno inseriti i caratteri semigrafici che si ottengono premendo gli stessi tasti, ma contemporaneamente ad uno dei due tasti Shift oppure al tastino Commodore (in basso a sinistra).

Non tutti sanno, però, che un qualsiasi simbolo alfa-numerico-grafico può esser visualizzato in altri due modi.

Il primo è quello di usare ancora il comando Print, ma relativo al codice del carattere individuato da un numero compreso tra 0 e 255.

Ad esempio, il comando...

Print Chr$(65); Chr$(214); Chr$(187)

...farà apparire una "A" maiuscola, una grossa "X" (che, però, è ben diversa dal carattere "X" maiuscolo) ed un quadratino.

Ai codici 65, 214 e 187, infatti, corrispondono, rispettivamente, i tre caratteri apparsi sul video.

Naturalmente se provate a premere insieme i tasti Shift e Commodore noterete che la "A" diventa minuscola, la "X" diventa una "V" maiuscola mentre il quadratino rimane immutato. Ciò è dovuto al fatto che i caratteri gestibili mediante il tasto Shift diventano minuscoli-maiuscoli oppure maiuscoli-grafici; quasi tutti i simboli generati con il tasto Commodore, al contrario, rimangono immutati.

Si noti che i codici da assegnare a Chr$() sono ben 256 (numerati da 0 a 255) ma solo alcuni sono realmente visibili; molti di loro, infatti, non hanno alcun significato pratico; altri sono i codici che consentono di cambiar colore, di attivare il Reverse e di controllare i tasti del cursore.

Un ciclo For..Next del tipo seguente...

For i=0 to 255: Print Chr$(i);: Next

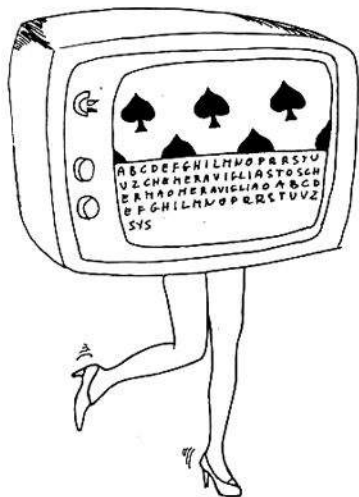
...sembra fornire risultati non "progressivi" proprio perchè i valori corrispondenti ai simboli visibili non sono posti in successione.

Uno sguardo al libretto di istruzioni del computer vi consentirà, comunque, di approfondire l'argomento.

Per conoscere il codice di un qualsiasi simbolo è utile la funzione Asc. Esempio:

Print Asc("R")

... permette di conoscere il valore del carattere racchiuso tra virgolette, in questo caso 210 ("R" maiuscolo).

Il sistema più immediato per far apparire immagini sul video è quello di ricorrere al comando Print

```
100 REM CARATTERI SEMI-GRAFICI
110 REM VISUALIZZATI CON PRINT
120 :
130 READ IA,OS,IB,VS,VD,OI
140 :
150 IA$=CHR$(IA):REM INCLINATA ALTO
160 OS$=CHR$(OS):REM ORIZZON. SUPER.
170 IB$=CHR$(IB):REM INCLINATA BASSO
180 VS$=CHR$(VS):REM VERTICALE SIN.
190 VD$=CHR$(VD):REM VERT. DESTRA
200 OI$=CHR$(OI):REM ORIZZ.INFERIORE
210 :
220 PRINTCHR$(147):REM CANC.SCHERMO
230 FOR I=1 TO 30
240 PRINT TAB(I)IA$;OS$;OS$;OS$;IB$
250 PRINT TAB(I)IB$;OI$;OI$;OI$;IA$
260 GOSUB 310: NEXT
270 FOR I=1 TO 30: X=30-I
280 PRINT TAB(X)IA$;OS$;OS$;OS$;IB$
290 PRINT TAB(X)IB$;OI$;OI$;OI$;IA$
300 GOSUB 310: NEXT: GOTO 230
310 FOR J=1 TO 100: NEXT
320 RETURN
330 :
340 DATA 206,163,205,165,167,164
```

Le funzioni Chr$() e Asc() sono utili per conoscere i codici dei vari caratteri

Il metodo Poke

Un altro sistema di visualizzazione è adoperato spesso per creare piccoli disegni ricorrendo, sempre, ai caratteri semigrafici.

Lo schermo dei piccoli computer Commodore (C/64, C/16, Plus/4, C/128) è suddiviso in 1000 celle (locazioni di memoria Ram) ognuna delle quali è individuabile dal numero del corrispondente indirizzo.

Nel caso del C/64 (e del C/128 in modo 40 colonne), ad esempio, la prima cella del video in alto a sinistra è quella n.1024, quella adiacente è la 1025 e, procedendo, l'ultima (quella, cioè, in basso a destra) la 2023.

Per il C/16 (e Plus/4) la prima cella è la n.3072, mentre l'ultima è numerata con 4071.

Poichè una qualsiasi locazione Ram può contenere un valore compreso tra 0 e 255, e può esser gestita mediante il comando Poke, sarà possibile far apparire tutti i caratteri mediante un ciclo For...Next molto semplice:

For X=0 to 255: Poke 1024+X,X: Next

Nel caso del C/16, ovviamente, sarà necessario scrivere 3072 al posto di 1024.

A differenza del ciclo For...Next relativo alla funzione Chr$() vista prima, in questo caso tutto sembra svolgersi "regolarmente" dal momento che i vari simboli appaiono in ordine l'uno dopo l'altro a partire dal bordo superiore sinistro.

I listati

I brevissimi listati, che consigliamo di studiare a fondo, permettono di esaminare alcune tecniche di programmazione legate sia alla prima che alla seconda tecnica descritta. Si noti come, nel primo caso, sia relativamente semplice associare, ad altrettante variabili stringa, i corrispondenti caratteri Ascii che si intende riportare su

```
100 REM CARATTERI SEMI-GRAFICI
110 REM VISUALIZZATI CON POKE
120 :
130 READ IA,OS,IB,VS,VD,OI
140 PRINT CHR$(147): X=1024:Y=X
150 REM X= 1024 SOLO PER C/64
160 REM PER C/16 DIGITARE X=3072
170 POKE X,IA: FOR I=X+1 TO X+10
180 POKE I,OS:NEXT:REM DISEGNO N.1
190 POKE X+11,IB: GOSUB 260
200 POKE Y,IB: FOR I=Y+1 TO Y+10
210 POKE I,OI:NEXT:REM DISEGNO N.2
220 POKE Y+11,IA:GOSUB 260
230 GOTO 170: REM RICOMINCIA
240 :
250 REM SUBROUTINE DI RITARDO
260 FOR J=1 TO 100: NEXT: RETURN
270 REM DATI PER POKE DI SCHERMO
280 DATA 78,99,77,101,103,100
```

schermo.

Una po' di buona volontà permette di utilizzare un vettore da gestire con maggior disinvoltura.

Il secondo listato, che riproduce un effetto lievemente diverso dal precedente, rappresenta un suggerimento per sviluppare tecniche di programmazione più sofisticate.

Il terzo listato (che funziona solo con il C/64) sfrutta un modo insolito il comando Input riferito allo schermo.

Dopo aver battuto Run (e premuto il tasto Return) premete i vari tasti, magari insieme ad uno dei tasti Shift oppure Commodore, facendo attenzione a non "scendere" troppo. Premendo, in qualsiasi momento, il tasto Return, il disegno realizzato verrà visualizzato nuovamente. I più bravini potranno aggiungere le migliorie del caso in modo da disegnare, memorizzare su supporto magnetico e richiamare le schermate semigrafiche realizzate.

L'indirizzo della prima cella video è diverso per il C/64 (1024) e per il C/16 (3072)

```
100 REM PROGRAMMA PER DISEGNARE
110 REM SCHERMATE SEMI-GRAFICHE
120 DIM A$(24):PRINTCHR$(147);
130 FOR I=1 TO 24
140 PRINT CHR$(165): NEXT
150 X$=CHR$(13):PRINTCHR$(19);
160 GET A$
170 IF A$=X$ THEN 190
180 PRINT A$;:GOTO 160
190 PRINT CHR$(19);:OPEN 1,3
200 FOR I=1 TO 24
210 INPUT#1,A$(I):NEXT:CLOSE1
220 PRINT"ECCO IL TUO DISEGNO"
230 GOSUB 260:PRINT CHR$(147);
240 FOR I=1 TO 24:PRINT A$(I)
250 NEXT:GOSUB 260: GOTO 220
260 GET A$: IF A$="" THEN 260
270 RETURN
```

La gestione del bordo fa parte delle tecniche di interruzione

Negli ultimi tempi si sono sviluppate tecniche di programmazione veramente incredibili per il C/64, tanto da rasentare l'impossibile.

Certamente a qualcuno di voi sarà capitato, almeno una volta, di osservare giochi o programmi che utilizzano messaggi o disegni posizionati al di fuori dallo schermo, cioè sul bordo dello stesso video.

Questo articolo è stato sviluppato per divulgare tali tecniche, in modo da offrire a chiunque la possibilità di applicarle in propri programmi.

E' bene affermare subito una nota importante, che non tutti conoscono: nel bordo possono essere visualizzati solo sprite. Per verificarlo, provate a digitare la seguente istruzione:

```
Poke 53269,255
```

Essa attiva gli otto sprite nello schermo, ma non riusciamo a vederli per il semplice fatto che sono posizionati, all'accensione della macchina, nell'angolo sinistro superiore, al di sotto del bordo.

Nella figura 1 sono indicati i valori delle posizioni degli sprite per cui questi risultano visibili nello schermo ed è anche rappresentato il bordo.

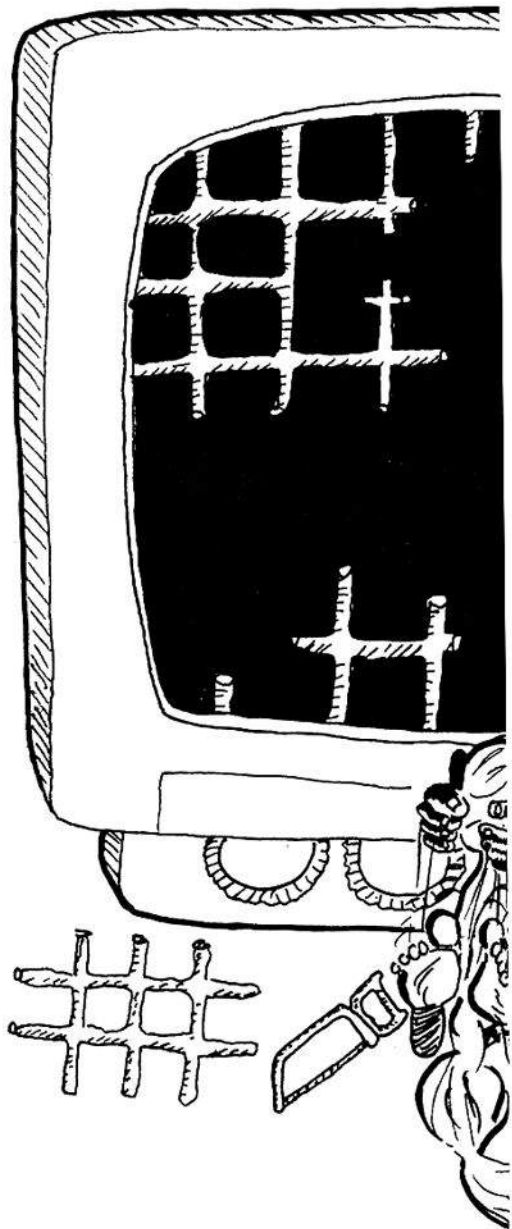
Ciò dimostra perchè possono essere visualizzati solo sprite; rimane ancora da capire come è possibile visualizzare gli sprite senza vedere il bordo.

D'ora in poi chiameremo "finestra sul bordo" quella porzione di spazio che, una volta "aperto" il bordo, visualizza gli sprite.

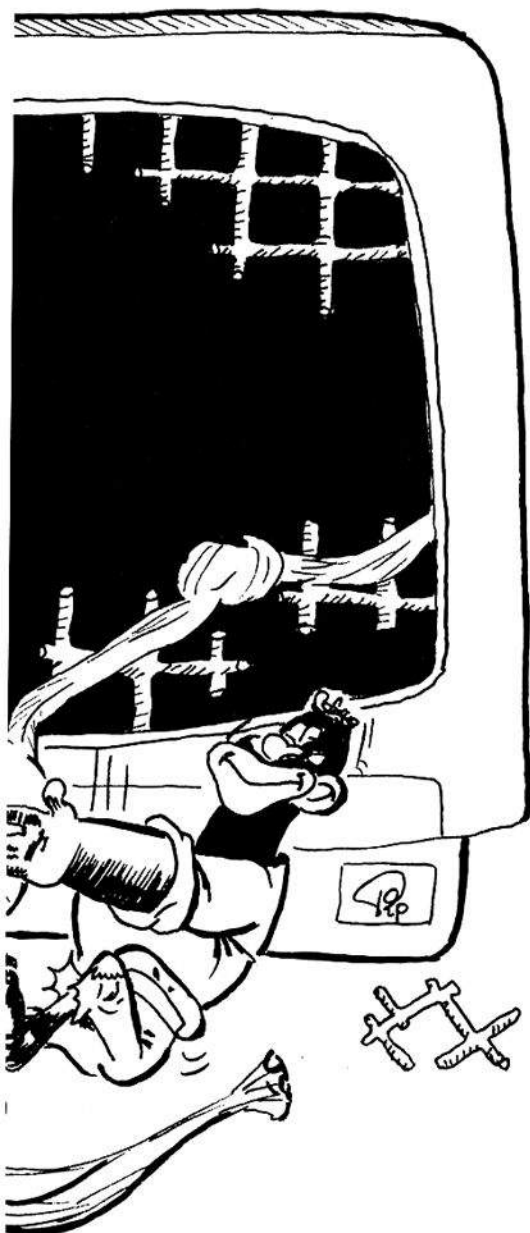
Per capire come sia possibile visualizzare la finestra, bisogna conoscere le specifiche del circuito integrato 6566/6567, il famoso VIC II.

Questo chip, presente sulla scheda del C/64, si occupa del controllo dei colori del video e dei caratteri, "vede" fino a 16 Kbytes di memoria per le informazioni relative al video (320*200 pixel) ed infine contiene, al suo interno, ben 47 registri di controllo, di cui diamo una completa descrizione nella figura 2.

Per realizzare la finestra sul bordo è necessario programmare alcuni di tali registri, i cui flag (=bit-interruttori) gestiscono anche il bordo.

E' proprio grazie ai flag, infatti, che vengono inviati gli opportuni segnali video (oscillatori verticali e orizzontali) creando l'immagine sullo schermo.

AL DI LA' DELLE S

Come far apparire gli sprite

di **Davide**

BARRE DEL VIDEO

del C/64 sul bordo del video

Ardizzone

Il linguaggio macchina è indispensabile per aprire "finestre" sui bordi

Prima di spiegare, almeno in parte, il loro funzionamento elettronico, parleremo di un particolare registro chiamato "Raster" (allocato in $D012).

Stiamo parlando di quel raggio luminoso che crea l'immagine in un televisore o in un monitor; il suo movimento è continuo, da sinistra verso destra e dall'alto verso il basso; una volta arrivato in basso a destra, ritorna in alto a sinistra, riprendendo l'operazione; questo fenomeno si ripete 25 volte al secondo.

Ritornando ai flag, possiamo dire che questi controllano la posizione del nostro Raster che, arrivato in determinate posizioni, a sua volta attiva o disattiva altri flag che formano il bordo.

Per aprire una finestra possiamo quindi ingannare il computer facendogli credere di aver già tracciato il bordo, mentre, in realtà, ciò non si è verificato; inoltre comandiamo il Raster per posizionare la finestra sul bordo in un punto a nostro piacere purchè vengano rispettati, ovviamente, i tempi di esecuzione per non disturbare il funzionamento del sistema operativo.

Ciò significa che, una volta aperta la finestra sul bordo, possiamo attivare nostri programmi Basic o l.m. senza disturbare la routine che si occupa della finestra e senza che questa disturbi il nostro programma.

Per operare correttamente, bisogna programmare un particolare Interrupt chiamato scansione dell'immagine.

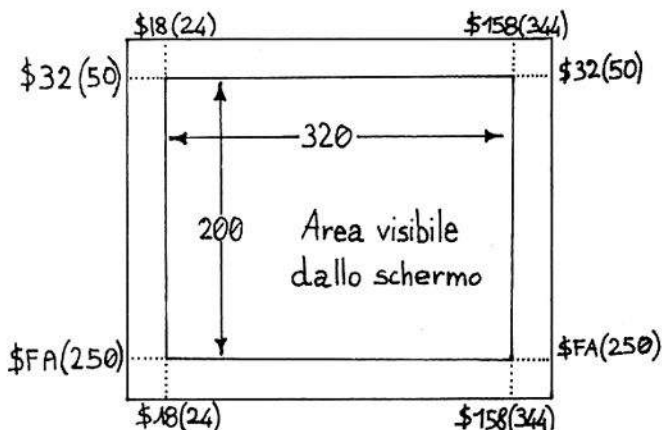
Una questione di IRQ

Che cosa siano gli Interrupt (che chiameremo d'ora in poi, semplicemente, IRQ) dovrebbe esser già noto, dal momento che questo articolo è destinato agli esperti. Tuttavia, per rinfrescare la memoria, ricorderemo alcune utili informazioni.

L'IRQ costituisce una funzione importante del microprocessore che svolge le procedure del sistema ogni sessantesimo di secondo; la sua gestione è agevole in quanto basta modificare un vettore di due bytes, (788-789 $0314-$0315) per fare in modo che nostre routine l.m. vengano eseguite nello stesso lasso di tempo.

Per "inserire" queste ultime nel ciclo normale, dovremo:

- Scrivere la nostra routine lm. in una parte libera della memoria Ram, tenendo presen-

Poter gestire sprite sul bordo consente di scrivere videogame altrimenti impossibili

te che l'ultima sua istruzione DEVE essere JMP $EA31, che provvede a svolgere gli indispensabili controlli di sistema.

• Disattivare ogni richiesta di Interrupt per evitare blocchi del sistema, usufruendo dell'istruzione SEI (set enable interrupt).

• Modificare il vettore IRQ, in modo che "punti" alla nostra routine.

• Attivare nuovamente l'IRQ originario, ricorrendo all'istruzione CLI (clear Irq).

La figura 3 mostra l'IRQ non modificato nella mappa di memoria, mentre la figura 4 mostra l'IRQ modificato.

Da questo momento distingueremo un IRQ "normale" dall'IRQ di scansione, che è quel particolare IRQ che permette di aprire una finestra sul bordo.

Esso controlla tre particolari registri:

• $D012 = raster (possiamo leggervi o scrivervi un dato)

• $D011 = bit 7 (che corrisponde al "nono" bit della locazione $d012)

• $D019: questa locazione corrisponde al flag dell'eventuale esecuzione di un IRQ, determinata più precisamente dal valore del bit meno significativo; se un IRQ è già avvenuto in precedenza, il bit zero è spento; dobbiamo quindi resettarlo mettendolo a uno, permettendo l'esecuzione del nostro IRQ.

• $D01A: per scrivere nel registro $D012, dobbiamo, prima di tutto, accendere il bit meno significativo di questo registro. Per concludere l'IRQ di scansione invece che usare l'istruzione JMP $EA31 facciamo uso di JMP $EA81 oppure JMP $FEBC.

• DC0E: attiva un timer per il nostro IRQ.

• DC0D: mettendo a zero il bit 7 di questo registro disabiliteremo ogni possibile IRQ e solamente il nostro verrà "servito" evitando

così tutte le interferenze tipiche di questa tecnica.

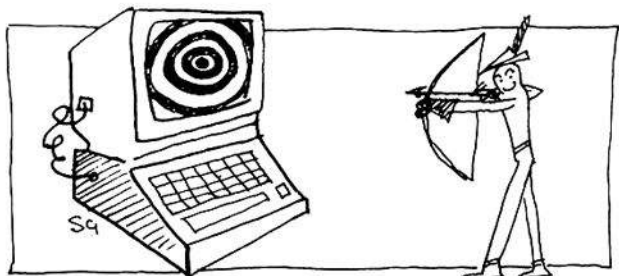
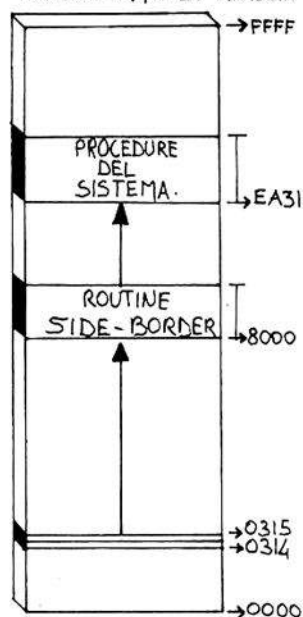
In poche parole l'IRQ di scansione permette di gestire il video controllando il pennello ed usufruendo dei registri; un "normale" IRQ, invece, si limita a svolgere altre funzioni, tra cui attivare i tasti di funzione, musiche e così via.

Il primo programma

Il primo programma di queste pagine (Side border 1) esegue l'apertura del bordo superiore ed inferiore dello schermo.

La sua esecuzione è molto semplice, tanto che può essere integrato facilmente nei vostri programmi senza determinare ritardi di nessun genere da parte del sistema.

Quando il raster raggiunge la posizione 250, vale a dire nel momento in cui sta per entrare nell'estremità inferiore del bordo, nel registro $D011 viene inserito forzatamente il valore 19 che impone, di solito, l'apertura del bordo stesso. Quando, in seguito, il raster raggiuge nuovamente la posizione 32, la locazione $D011 riaquista il suo valore standard.

VEDUTA DEL IRQ MODIFICATO
NELLA MAPPA DI MEMORIA

In questa porzione di schermo possono essere visualizzati sprite fermi o in movimento, da gestire per mezzo di comandi Basic Poke.

Side-border 2

Il secondo programma provvede ad aprire una finestra nel bordo laterale della stessa grandezza degli sprite che devono essere in fila uno di seguito all'altro.

Le ordinate degli sprite devono avere il valore identico alla posizione dell'apertura del bordo gestita dal raster; in caso contrario il bordo si chiude.

La finestra laterale può essere inserita dalla posizione (verticale) 50 alla 250, controllata dal raster.

Nel listato Basic alla riga 150 possiamo cambiare il valore della variabile "P" che indica la posizione della finestra e degli sprite sul video (da 50 a 250).

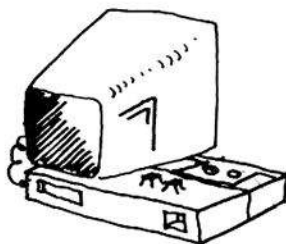
Anche questo listato può essere facilmente integrato nei vostri programmi, purchè si presti attenzione al fatto che questa finestra sposta in basso di tre righe il testo facendole sparire completamente.

Un (piccolo) difetto del programma è che rallenta le esecuzioni del sistema, a causa degli indispensabili controlli per non avere continue interferenze sul bordo (compresi tremolii di varia natura).

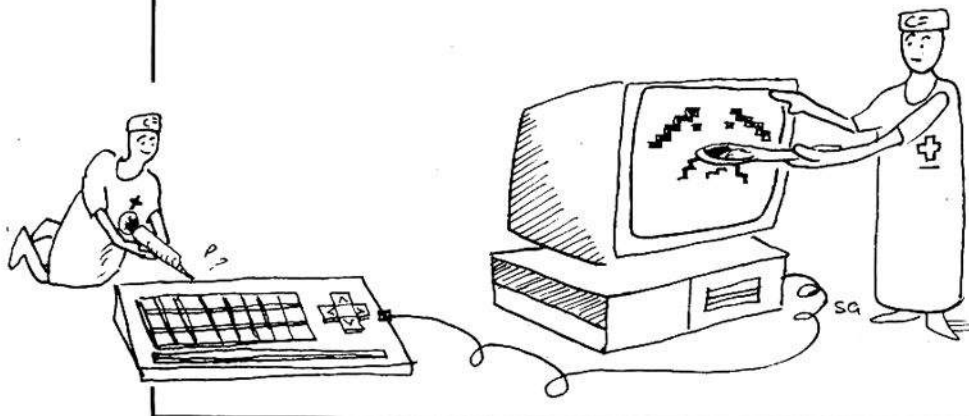
Il funzionamento del programma è di memorizzare la posizione della finestra nel raster e di incrementare e decrementanre il registro $D016 gestito da un perfetto controllo dello schermo e da una quasi perfetta temporizzazione che permette la visualizzazione della finestra laterale sul video.

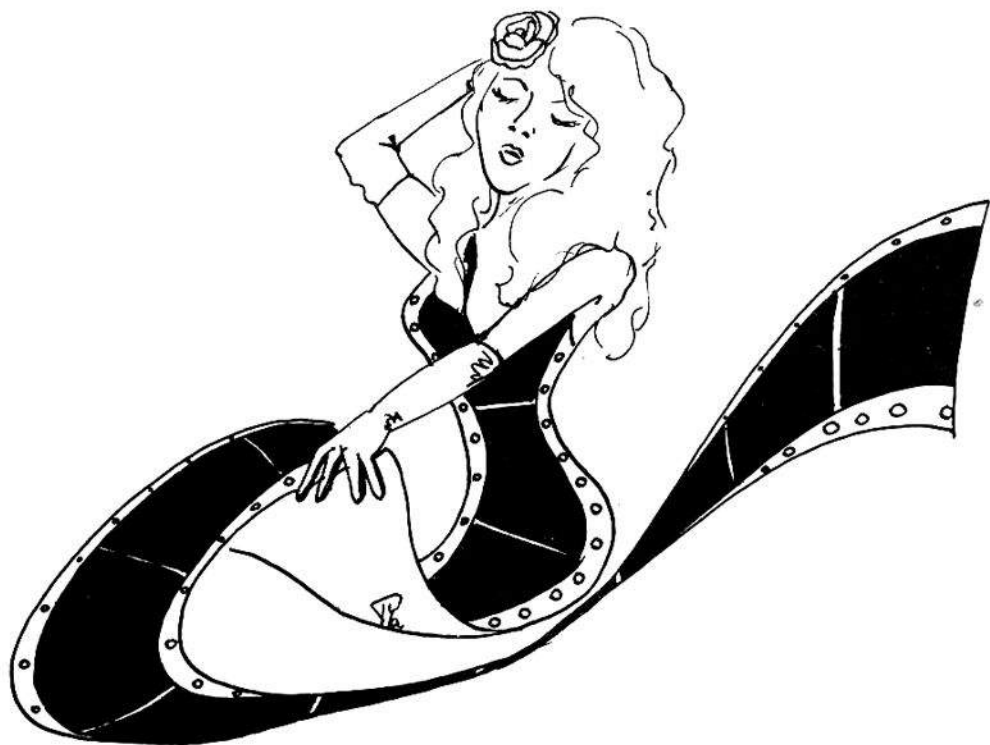
Coloro che desiderino comprendere a fondo la tecnica di interruzione, possono far riferimento ai disassemblati commentati.

I programmi proposti sono brevissimi: digitateli con attenzione

```
Controllore interfaccia video 6566/6567 (VIC II) .
Indirizzi di memoria.   HEX$--DEC.
D000--53248 :ASCISSA  DELLO SPRITE 0.
D001--53249 :ORDINATA DELLO SPRITE 0.
D002--53250 :ASCISSA  DELLO SPRITE 1.
D003--53251 :ORDINATA DELLO SPRITE 1.
D004--53252 :ASCISSA  DELLO SPRITE 2.
D005--53253 :ORDINATA DELLO SPRITE 2.
D006--53254 :ASCISSA  DELLO SPRITE 3.
D007--53255 :ORDINATA DELLO SPRITE 3.
D008--53256 :ASCISSA  DELLO SPRITE 4.
D009--53257 :ORDINATA DELLO SPRITE 4.
D00A--53258 :ASCISSA  DELLO SPRITE 5.
D00B--53259 :ORDINATA DELLO SPRITE 5.
D00C--53260 :ASCISSA  DELLO SPRITE 6.
D00D--53261 :ORDINATA DELLO SPRITE 6.
D00E--53262 :ASCISSA  DELLO SPRITE 7.
D00F--53263 :ORDINATA DELLO SPRITE 7.
D010--53264 :BITS PIU' SIGNIFICATIVI DELLE ASCISSE DEI OTTO SPRITE,
             AD OGNUNO DI ESSI CORRISPONDE IL BIT PROPRIO.
D011--53265 :REGISTRO DI CONTROLLO DEL VIC 6566/6567.
             (7) COMPARATORE DI QUADRO.
             (6) EXTENDED COLOR MODE (1=ABILITATO).
             (5) ALTA RISOLUZIONE    (1=ABILITATA).
             (4) RIEMPIE LO SCHERMO CON IL COLORE DEL BORDO.
             (3) SELEZIONE LINEE DI TESTO VIDEO (24 O 25).
             (2-1-0) SCROLLING DI UN ORDINATA DI UN PUNTO (0-7)
D012--53266 :LETTURA O SCRITTURA DEL PENNELLO OTTICO.
D013--53267 :ASCISSA POSIZIONE PENNA OTTICA.
D014--53268 :ORDINATA POSIZIONE PENNA OTTICA.
D015--53269 :ABILITATORE SPRITES (1=ABILITATO)
D016--53270 :(7-6) NON USATI.
             (5) QUESTO BIT DEVE ESSERE SEMPRE SPENTO.
             (4) MODO MULTICOLOR (1=ABILITATO).
             (3) SELEZIONA COLONNE DEL TESTO VIDEO (38 O 40).
             (2-1-0) SCROLLING DELL'ASCISSE DI UN PUNTO (0-7).
D017--53271 :ESPANSIONE VERTICALE DEL DOPPIO DEGLI SPRITES.
D018--53272 :REGISTRO DI CONTROLLO DELLA MEMORIA DEL VIC 6566/6567.
             (7-6-5-4) INDIRIZZO BASE DELLA MATRICE DEL VIDEO.
             (3-2-1-0) INDIRIZZO BASE DEL PUNTO DI UN CARATTERE.
D019--53273 :REGISTRO CONTROLLO FLAGS DI IRQ.
D01A--53274 :REGISTRO MASCHERA IRQ.
D01B--53275 :PRIORITA' TRA SPRITE E SFONDO (1=SPRITE).
D01C--53276 :MODO MULTICOLOR PER GLI SPRITES (1=ABILITATO).
D01D--53277 :ESPANSIONE ORIZZONTALE DEL DOPPIO DEGLI SPRITES.
D01E--53278 :REGISTRO RIVELAZIONE DI CONTATTO TRA DUE SPRITES.
D01F--53279 :REGISTRO RIVELAZIONE DI CONTATTO TRA SPRITES E SFONDO.
D020--53280 :COLORE DEL BORDO.
D021--53281 :COLORE DELLA PAGINA SCHERMO.
D022--53282 :COLORE 1 DELLO SFONDO.
D023--53283 :COLORE 2 DELLO SFONDO.
D024--53284 :COLORE 3 DELLO SFONDO.
D025--53285 :COLORE MULTICOLOR 0 PER SPRITES.
D026--53286 :COLORE MULTICOLOR 1 PER SPRITES.
D027--53287 :COLORE DELLO SPRITE 0.
D028--53288 :COLORE DELLO SPRITE 1.
D029--53289 :COLORE DELLO SPRITE 2.
D02A--53290 :COLORE DELLO SPRITE 3.
D02B--53291 :COLORE DELLO SPRITE 4.
D02C--53292 :COLORE DELLO SPRITE 5.
D02D--53293 :COLORE DELLO SPRITE 6.
D02E--53294 :COLORE DELLO SPRITE 7.
```

```
1Ø REM                          LISTATO SOLO PER C-64
2Ø REM ***************************************************************
3Ø REM                                SIDE BORDER 1
4Ø REM                           BY DAVIDE ARDIZZONE
5Ø REM ***************************************************************
1ØØ PRINT"SIDE-BORDER 1":POKE53280,Ø:POKE53281,Ø:
11Ø FORI=1TO8Ø
115 READA:IF A>255 OR A<Ø THEN PRINT"ERRORE NEI DATA":LIST2ØØ-
12Ø POKE 32767+I,A:PRINT"DATO=";I;"":B=B+A:NEXT
13Ø IF B<>9787 THEN PRINT "ERRORE NEI DATA.":LIST2ØØ-
14Ø PRINT"DATI O.K."
15Ø POKE 53269,255:POKE53277,255:POKE53271,255:POKE53264,7:C=32:D=Ø
16Ø FORI=ØTO7:POKE53287+I,7:POKE2Ø4Ø+I,C:C=C+1:NEXT
17Ø FORI=1TO16STEP2:POKE53249+I,Ø:NEXT
18Ø FORI=ØTO15STEP2:POKE53248+I,D:D=D+48:IF D>255 THEN D=D-255
185 NEXTI
19Ø SYS 3276Ø
2ØØ DATA12Ø,169,47,162,128,141,2Ø,3,142,21,3,169,128,141,18,2Ø8
21Ø DATA173,17,2Ø8,41,127,141,17,2Ø8,169,24Ø,141,26,2Ø8,169,1,141
22Ø DATA14,22Ø,169,2,141,32,2Ø8,169,Ø,141,33,2Ø8,88,96,234,169
23Ø DATA1,141,25,2Ø8,169,248,2Ø5,18,2Ø8,2Ø8,251,169,19,141,17,2Ø8
24Ø DATA169,ØØ,2Ø5,18,2Ø8,2Ø8,251,169,27,141,17,2Ø8,76,49,234,Ø
1ØØØ FORJ=ØTO7:FORI=ØTO255:POKE53249+J,I:NEXTI,J
```

```
1Ø REM                          LISTATO SOLO PER C-64
2Ø REM ***************************************************************
3Ø REM                                SIDE BORDER 2
4Ø REM                           BY DAVIDE ARDIZZONE
5Ø REM ***************************************************************
1ØØ PRINT"SIDE-BORDER 2":POKE53280,Ø:POKE53281,Ø:
11Ø FORI=1TO23Ø
115 READA:IF A>255 OR A<Ø THEN PRINT"ERRORE NEI DATA"
12Ø POKE 32767+I,A:PRINT"DATO=";I;"":B=B+A:NEXT
13Ø IF B<>27669 THEN PRINT "ERRORE NEI DATA."
14Ø PRINT"DATI O.K."
15Ø P=128:REM POSIZIONE APERTURA NEL BORDO
16Ø POKE 32825,P:POKE329Ø8,P
17Ø SYS 32768:REM $8ØØØ
199 END
2ØØ DATA12Ø,169,124,141,2Ø,3,169,128,141,21,3,169,Ø,141,18,2Ø8
21Ø DATA173,17,2Ø8,41,127,141,17,2Ø8,169,1,141,26,2Ø8,169,Ø,141
22Ø DATA14,22Ø,88,169,255,141,21,2Ø8,141,29,2Ø8,169,193,141,16,2Ø8
23Ø DATA169,Ø,141,32,2Ø8,141,33,2Ø8,169,128,162,1,157,Ø,2Ø8,232
24Ø DATA232,224,17,2Ø8,247,169,24Ø,141,Ø,2Ø8,169,4Ø,141,2,2Ø8,169
25Ø DATA88,141,4,2Ø8,169,136,141,6,2Ø8,169,184,141,8,2Ø8,169,232
26Ø DATA141,1Ø,2Ø8,169,24,141,12,2Ø8,169,72,141,14,2Ø8,162,Ø,16Ø
27Ø DATA32,152,157,248,7,2ØØ,232,224,8,2Ø8,246,96,169,1,141,25
28Ø DATA2Ø8,169,147,141,2Ø,3,169,128,141,21,3,169,128,141,18,2Ø8
29Ø DATA76,129,234,162,21,173,18,2Ø8,24,1Ø5,5,41,7,9,24,141
3ØØ DATA17,2Ø8,16Ø,6,136,2Ø8,253,36,1,2Ø6,22,2Ø8,238,22,2Ø8,36
31Ø DATA1,234,234,36,1,234,234,234,234,173,18,2Ø8,24,1Ø5,5,41
32Ø DATA7,9,24,141,17,2Ø8,2Ø6,22,2Ø8,238,22,2Ø8,2Ø2,2Ø8,227,169
33Ø DATA1,141,25,2Ø8,169,124,141,2Ø,3,169,128,141,21,3,169,8
34Ø DATA141,18,2Ø8,76,49,234
35Ø REM ****** FINE. ******
```

```
 1 **** LISTATO SOLO PER C-64 ****
 2 *******************************
 3 * QUESTO PROG. , GESTITO DA UN *
 4 * IRQ, APRE UNA FINESTRA NEL   *
 5 * BORDO LATERALE DELLO SCHERMO:*
 6 * PRECISAMENTE DALLA POSIZIONE *
 7 * CONTROLLATA DAL RASTER CHE VA*
 8 * DA $32(#50) ALLA $F8(#248).  *
 9 * IN QUESTA FINESTRA POSSONO   *
10 * ESSERE VISUALIZZATI SOLAMENTE*
11 * DEGLI SPRITE.                *
12 *******************************
13 *         SIDE BORDER 2        *
14 *       BY DAVIDE ARDIZZONE    *
15 *     DISASSEMBLATO PER C-64.  *
16 *******************************
17 *       INIZIO PROGRAMMA
18 *******************************
19 RASTER    =      $D012       ;PENNELLO OTTICO(RASTER).
20 POSSPR    =      $80         ;POSIZIONE DELL'APERTURA
21                               ;SUL BORDO E POSIZIONE
22                               ;DEGLI SPRITE.
23 *******************************
24 START      SEI                ;SETTA IRQ A 1.
25            LDA    #<IRQ1      ;BYTE BASSO.
26            STA    $0314       ;VETTORE BASSO IRQ.
27            LDA    #>IRQ1      ;BYTE ALTO.
28            STA    $0315       ;VETTORE ALTO IRQ.
29            LDA    #00         ;IL PENNELLO OTTICO
30            STA    RASTER      ;ASSUME VALORE 0.
31            LDA    $D011       ;AZZERA IL
32            AND    #$7F        ;COMPARATORE
33            STA    $D011       ;DI QUADRO.
34            LDA    #01         ;ABILITA I CONTROLLI
35            STA    $D01A       ;NECESSARI AL NOSTRO IRQ.
36            LDA    #00         ;DISABILITA IL TIMER 'A'
37            STA    $DC0E       ;DEL CIA #1.
38            CLI                ;ATTIVA IRQ.
39            LDA    #$FF        ;ATTIVA GLI OTTO
40            STA    $D015       ;SPRITE DISPONIBILI.
41            STA    $D01D       ;ESPANSIONE X SPRITE.
42            LDA    #$C1        ;BYTE ALTO PER LA
43            STA    $D010       ;POSIZIONE DEGLI SPRITE.
```

```
44            LDA    #00         ;COLORE NERO
45            STA    $D020       ;NEL BORDO
46            STA    $D021       ;E NELLA PAGINA.
47            LDA    #POSSPR     ;POSIZIONA
48            LDX    #01         ;LE ORDINATE
49 SPR1       STA    $D000,X     ;DEGLI OTTO
50            INX                ;SPRITE
51            INX                ;NELLO
52            CPX    #$11        ;SCHERMO.
53            BNE    SPR1        ;
54            LDA    #$F0        ;POSIZIONA
55            STA    $D000       ;LE ASCISSE
56            LDA    #$28        ;DEGLI
57            STA    $D002       ;OTTO
58            LDA    #$58        ;SPRITE
59            STA    $D004       ;NELLO
60            LDA    #$88        ;SCHERMO.
61            STA    $D006       ;
62            LDA    #$B8        ;
63            STA    $D008       ;
64            LDA    #$E8        ;
65            STA    $D00A       ;
66            LDA    #$18        ;
67            STA    $D00C       ;
68            LDA    #$48        ;
69            STA    $D00E       ;
70            LDX    #$00        ;LETTURA DELLA
71            LDY    #$20        ;RIDEFINIZIONE
72 SPR2       TYA                ;GRAFICA DEGLI
73            STA    $07F8,X     ;OTTO SPRITE,
74            INY                ;ALL'INTERNO
75            INX                ;DELLA
76            CPX    #08         ;MEMORIA.
77            BNE    SPR2        ;
78            RTS                ;RITORNO AL BASIC.
79 ********************************
80 * ROUTINE IRQ PRINCIPALE..
81 ********************************
82 IRQ1       LDA    #01         ;ATTIVA CONTROLLO TRA
```

```
 83            STA   $D019       ;SPRITE E BORDO.
 84            LDA   #<IRQ2      ;PUNTA L'IRQ
 85            STA   $0314       ;IN UNA
 86            LDA   #>IRQ2      ;SECONDA
 87            STA   $0315       ;ROUTINE.
 88            LDA   #POSSPR     ;VALORE PER L'APERTURA
 89                              ;DEL BORDO, INSERITO
 90            STA   RASTER      ;NEL REGISTRO DEL RASTER.
 91            JMP   $EA81       ;ESEGUE LA SECONDA
 92                              ;ROUTINE.
 93 ********************************
 94 * ROUTINE IRQ SECONDARIA.....
 95 ********************************
 96 IRQ2       LDX   #$15        ;NUMERO DI CICLI PER
 97                              ;L'APERTURA DEL BORDO.
 98            LDA   RASTER      ;LEGGE LA POSIZIONE DEL
 99            CLC               ;RASTER PER SPOSTARE LA
100            ADC   #05         ;PAGINA SCHERMO UN PO'
101            AND   #07         ;SOTTO, IN MODO CHE IL
102            ORA   #$18        ;CURSORE NON DISTURBI LA
103            STA   $D011       ;FINESTRA SUL BORDO.
104            LDY   #06         ;PICCOLO
105 RIT1       DEY               ;RITARDO.
106            BNE   RIT1        ;
107            BIT   01          ;
108            DEC   $D016       ;QUESTO REGISTRO
109            INC   $D016       ;INCREMENTATO
110            BIT   01          ;E DECREMENTATO,
111            NOP               ;INSIEME A QUESTE
112 RIT2       NOP               ;ISTRUZIONI DI UN CERTO
113            BIT   01          ;RITARDO CALCOLATO,
114            NOP               ;DETERMINANO
115            NOP               ;L'APERTURA
116            NOP               ;DEL BORDO.
117            NOP               ;
118            LDA   RASTER      ;ESECUZIONE
119            CLC               ;IDENTICA
120            ADC   #05         ;A QUELLA
121            AND   #07         ;PRECEDENTE,
122            ORA   #$18        ;CON L'APERTURA
123            STA   $D011       ;DEL BORDO LATERALE,
124            DEC   $D016       ;SENZA
125            INC   $D016       ;DISTURBI.
126            DEX               ;RIPETE
127            BNE   RIT2        ;21 VOLTE.
128            LDA   #01         ;RIATTIVA IL CONTROLLO
129            STA   $D019       ;TRA SPRITE E BORDO.
130            LDA   #<IRQ1      ;PUNTA L'IRQ
131            STA   $0314       ;ALLA ROUTINE
132            LDA   #>IRQ1      ;PRINCIPALE.
133            STA   $0315       ;
134            LDA   #08         ;POSIZIONA IL
135            STA   RASTER      ;PENNELLO OTTICO.
136            JMP   $EA31       ;ESEGUE LE PROCEDURE
137 ********** FINE **********;NORMALI DEL SISTEMA.
```

MOVIE MAKER, SECONDO TEMPO

Come utilizzare i fotogrammi generati con "Movie Maker" in qualsiasi vostro programma

di **Valentino Spataro**

E' probabile che chi legge queste pagine abbia già letto l'articolo "Movie Maker" (C.C.C. n.51) oppure non ha la minima idea di cosa possa essere il programma, a parte quanto riesce a intuire dallo stesso nome.

Consigliamo, pertanto, almeno la lettura del riquadro, prima di continuare.

"Movie Maker" presenta due limiti: per realizzare un programma che si servisse dei fotogrammi prodotti è necessaria una certa esperienza (per quanto fosse stato descritto dettagliatamente il funzionamento del programma); inoltre, per rivedere il "filmino" prodotto, è necessario disporre in ogni caso delle routine di Toma, in quanto le routines l.m. del programma richiamano parte delle prime.

Il programma di queste pagine pone ri-

Disporre di brevi animazioni può conferire un aspetto originale ai vostri programmi.

La tecnica di visualizzazione della sequenza è totalmente automatica

medio ai due problemi: infatti svolge la funzione di riprodurre in sequenza i fotogrammi (per cui non dovrete spremere le meningi per scrivere un programma apposito), è stato predisposto per essere inserito in qualsiasi programma a vostra disposizione e funziona con qualsiasi espansione Basic che gestisca la pagina grafica.

Inoltre la routine l.m. è stata modificata affinchè possa funzionare indipendentemente dalle routine grafiche di Toma. Queste, tuttavia, devono essere utilizzate nella prima fase, cioè per "produrre" i singoli fotogrammi mediante "Movie Maker".

Come usare il programma

Dato il Run, ed atteso il caricamento della routine in l.m, verranno richiesti i nomi dei file contenenti i dati dei fotogrammi da caricare (precedentemente registrati grazie al programma di C.C.C. n.51); potrete immettere un massimo di 100 files, ma potete cambiare questo numero a piacere variando il valore della variabile NFILE in linea 60020.

Per avvertire il programma che la lista dei files è finita, alla ennesima richiesta del nome del file premete return "a vuoto" (senza, cioè, scrivere niente).

A questo punto il programma provvederà a caricare tutti i file indicando, di volta in volta, il nome del file, la memoria rimanente, il numero di bytes che restano da caricare da quel file ed il numero di file che avanzano.

Qualora qualcosa non funzionasse durante questa fase, il computer emette un messaggio di errore e passa al file successivo (senza quindi caricare quello per cui l'errore si è verificato); controlla anche di volta in volta se resta memoria Ram a disposizione: in caso negativo non carica più i files successivi. Poichè questa fase richiede tempo, potete aggiungere, prima del Goto 20 in linea 60110 (che fa tornare all'esecuzione del vostro programma) una routine che produca un beep in modo tale da avvertirvi del termine delle operazioni se vi siete allontanati dal computer.

A questo punto torna il controllo al vostro programma. Per vedere la sequenza di fotogrammi basterà fare un GOSUB 50000 dopo aver impostato X, Y, RT, NFOT(NFOT).

Le variabili X (compresa tra 0 e 39) ed Y (tra 0 e 24) indicano le coordinate dell'angolo superiore sinistro, RT indica il ritardo desiderato tra un fotogramma e l'altro (tramite un semplice ciclo FOR NEXT). Naturalmente le coordinate non devono essere in contraddizione con le dimensioni dei foto-

Che cos'è Movie Maker

"Movie Maker" è un editor grafico (pubblicato nel n.51 di C.C.C.) che permette di memorizzare grandi quantità di figure che vengono poi visualizzate in sequenza: per questo motivo le figure vengono chiamate, a buon diritto, "fotogrammi".

La possibilità di memorizzare un gran numero di fotogrammi contemporaneamente deriva del fatto che la routine l.m. per memorizzare i grafici può sfruttare anche gli 8K di memoria RAM nascosta sotto la ROM del Basic.

Inoltre, grazie alla generale prevalenza degli spazi vuoti presenti in una figura, la routine memorizza solo i bytes diversi da 0, mentre segna il numero di bytes nulli presenti dopo l'ultimo byte diverso da zero, con notevole risparmio di memoria. Supponiamo, ad esempio, che la seguente sia una sequenza di bytes che compone i dati di un fotogramma:

3, 45, 23, 2, 0, 23, 1

Essa indica che i primi 4 bytes della figura contengono bit accesi e spenti, rispettivamente, tanto da formare i valori 3, 45, 23 e 2 mentre i 23+1*256 bytes successivi sono uguali a zero.

Un eventuale numero zero, insomma, indica che i due bytes successivi si devono leggere nella forma low-byte e high-byte ad indicare il numero di bytes uguali a zero che seguono.

grammi memorizzati; se, quindi, non appare nulla, provate a diminuire il valore delle variabili X ed Y.

NFOT indica il numero di fotogrammi del filmino che vedremo e NFOT() contiene la sequenza dei fotogrammi da visualizzare in accordo con il seguente "codice"...

NFOT(1)=1; NFOT(2)=3; NFOT(3)=4; NFOT(4)=2; NFOT(5)=3;

...che, nell'ipotesi di aver caricato solo quattro fotogrammi, significherebbe: visualizza dapprima il fotogramma n.1, poi il n.3, poi il n.4, poi il n.2 e infine di nuovo il n.3.

Impostando adeguatamente NFOT() potete quindi vedere i fotogrammi in una sequenza o progressiva, o regressiva, o entrambe le cose insieme. NFOT è maggiore di NFILE proprio perchè è possibile rivedere più volte lo stesso fotogramma; NFOT è fissato ad un massimo di 300 in linea 60020, ma anche questo potete cambiarlo a piacere, compatibilmente con la memoria disponibile.

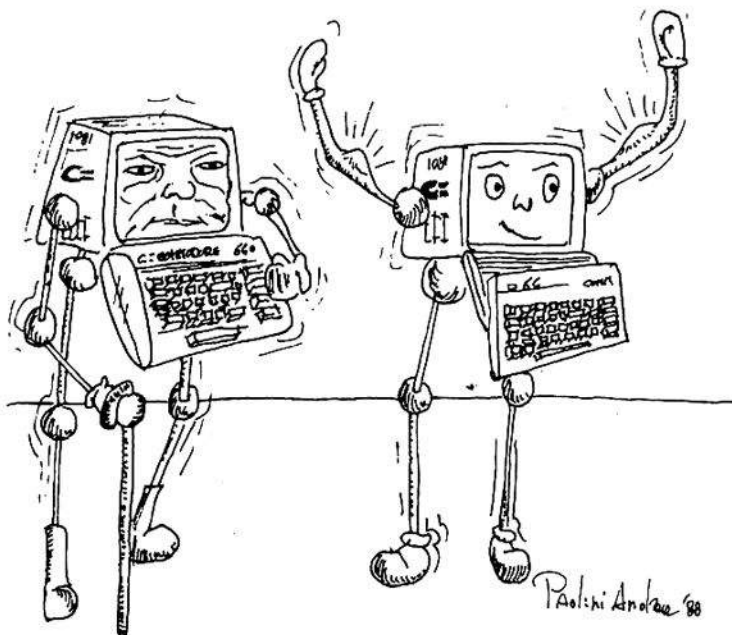
Per quanto riguarda X, Y, RT si consigliano, inizialmente, i seguenti valori:

1, 1, 300

Alla fine della visualizzazione premete un tasto qualsiasi e il programma, per come è fatto, chiederà nuovamente di immettere altri valori di X, Y, RT. Per fermare usate un brutale Run-Stop e Restore.

Un'ultima precisazione: caricate precedentemente le routine di Toma, il GW-BASIC oppure una qualsiasi altra espansione Basic, ed impostate di conseguenza le istruzioni per la pagina grafica in linea 30 e 35, che sono le uniche righe che contengono istruzioni non standard Commodore.

I controlli

I controlli sono numerosissimi: in fase di caricamento il programma controlla l'esistenza del file sul drive; può essere utile, a tal proposito, aggiungere un comando per vedere la directory in fase di input dei nomi dei files, cosa facilissima con il GW-BASIC avendo un'istruzione dedicata. Nel caso in cui qualcosa di negativo dovesse capitare, il

file non viene caricato. Viene sempre effettuato un controllo sulla memoria a disposizione e, in caso negativo, nessun altro file viene caricato.

Le dimensioni dei fotogrammi possono non essere uguali: può capitare, quindi, che alcuni fotogrammi stiano nei limiti dello schermo, ma altri no: per questo il programma controlla di volta in volta le coordinate X e Y a cui somma le dimensioni del fotogramma: nel caso in cui le dimensioni eccedano lo schermo, il programma salta al fotogramma successivo lasciando comunque attivo il ritardo in modo tale da accorgersi che qualcosa è stato saltato.

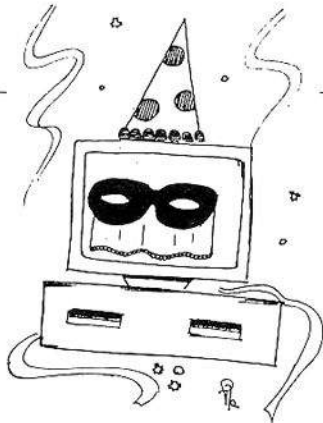
Sempre a causa dell'eventuale diversità delle dimensioni, può capitare che i fotogrammi non si sovrappongano completamente, lasciando sullo schermo parti del fotogramma precedente.

Occorrerà quindi aggiungere una routine che cancelli il fotogramma non appena sia trascorso il ritardo impostato per riportare la situazione alla normalità.

Facciamo un esempio: date le coordinate X e Y dell'angolo superiore sinistro, ed LX e LY le dimensioni del fotogramma, la routine per Toma potrebbe essere...

...mentre con il GW-BASIC sarebbe:

La routine l.m. è interamente rilocabile: basta modificare l'indirizzo iniziale

Sono necessarie le routine grafiche di Toma per generare i singoli fotogrammi

100 line (x*8,y*8)-((x+lx)* 8,(y+ly)*8),0,bf

100 a$="": For A=1 to Lx: a$=a$+ chr$(32): Next
110 For a=y to y+ly: char x,a,0,a$: Next

Le variabili

Nei vostri programmi, ai quali aggiungerete quello pubblicato in queste pagine, non dovrete usare i seguenti nomi di variabili...

NFOT: indica il numero di fotogrammi che compongono il filmino;
V: indica l'inizio della locazione di memoria a partire dalla quale si visualizza il grafico;
A, B: usate per due cicli di For...Next;
X, Y: indicano le coordinate dell'angolo superiore sinisto;
RT: indica il ritardo tra un fotogramma e l'altro;
LOC: indica lo start address della routine l.m.
...ed i vettori...
LOC(): indica alla routine l.m. a partire da quale locazione sono memorizzati i dati di quel fotogramma;
LX(), LY(): forniscono alla routine l.m. le dimensioni del fotogramma;
NFOT(): contiene la sequenza dei fotogrammi.

Il programma usa anche altre variabili, ma sono utilizzate solo in fase di inizializzazione: potete quindi riutilizzarle (previo azzeramento) nel vostro programma; fate attenzione a non "re-DIM-are" i vettori, tenendo conto che, oltre a quelli già citati, è presente FL$() che contiene i nomi dei files da caricare.

Le modifiche

Il programma, oltre alla possibilità di essere utilizzato con qualsiasi espansione basic, può anche essere usato con qualsiasi pagina grafica: in linea 50010, per esempio, il valore 57344 indica l'inizio della pagina grafica per Toma, GW-BASIC, SIMON'S BASIC. In altri casi esso è 8192. Modificate quindi tale valore a seconda delle vostre esigenze.

Inoltre la routine l.m. è allocata da 820 a 1020, ma può essere facilmente spostata altrove variando semplicemente il valore della variabile LOC in linea 60040 essendo INTERAMENTE RILOCABILE (Ooooh! come direbbero quelli di Indietro tutta).

Dulcis in fundo, la stessa routine è modificabile, come suggerito dalle REM nelle linee DATA: nella versione originale visualizza il grafico cancellando l'area sottostante, mentre nella versione modificata opera un "merge" con il disegno eventualmente già presente nella corrispondente porzione di schermo. Naturalmente, in quest'ultimo caso, i fotogrammi si sovrapporranno l'un l'altro.

Sul dischetto DIRECTORY di questo mese troverete, oltre al programma, anche alcuni fotogrammi.

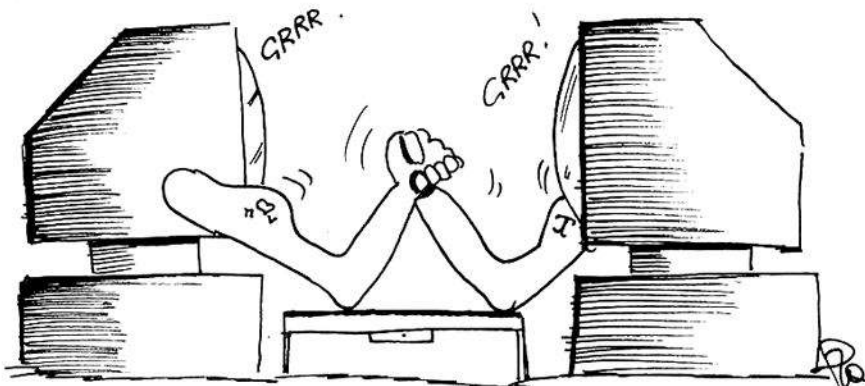
La memoria con "Movie Maker" II

Come nella prima versione, la seconda rimaneggia la memoria per avere a disposizione la RAM in cui memorizzare i fotogrammi. La mappa della memoria prende questa configurazione:

0-2047 :come all'accensione
2048-XXXX: programma pubblicato e programma utente in cui viene inserito.
XXXX-(XXXX+64*256): spazio per le variabili del programma
(XXXX+64*256)-LIM: spazio riservato per i dati dei fotogrammi
40720-40921: l.m. routine, peraltro rilocabile. Con GW-BASIC si consiglia da 820 a 1020.

l'indirizzo XXXX è indefinito: "Movie Maker" II lo determina, e ad esso si adegua: si può quindi modificare il programma a piacere; 64 è un valore che può essere variato ricordandosi di riservare RAM per i fotogrammi (vedi linea 60010); LIM (linea 60040) indica il TOP RAM e varia a seconda dell'espansione usata: da basic o da Toma è 40960, da GW-BASIC è 28672, da Simon's Basic 32748. LIM non può esser calcolata dal programma, ma deve essere inserita dall'utente.

```
4 REM ********************************
5 REM * MOVIE MAKER II BY SPATARO    *
6 REM * PER USARE I GRAFICI PRODOTTI *
7 REM * CON MOVIE MAKER IN QUALSIASI *
8 REM * PROGRAMMA                    *
9 REM ********************************
10 GOTO60000
20 REM PROGRAMMA IN CUI AGGIUNGERE LA SUBROUTINE: NON CANCELLARE QUESTA LINEA
30 ←TEXT0,1:INPUT"X,Y,RITARDO";X,Y,RT:REM IN GWBASIC SCREEN0
35 ←GRAF0,1:←CLEAR:REM IN GWBASIC SCREEN1:CLS1
40 FORA=1TOW:NFOT(A)=A:NEXT
45 GOSUB50000
49 REM PROGRAMMA IN CUI AGGIUNGERE LA SUBROUTINE
50 GETA$:IFA$=""GOTO50
55 GOTO30
60 :
65 REM ********************************
70 :
50000 REM PER VEDERE LA SEQUENZA IMPOSTARE X,Y,RT,NFOT()
50010 V=57344+X*8+Y*320:FORA=1TOW:IFX+LX(A)>39ORY+LY(A)>24GOTO50040
50020 POKE3,V/256:POKE2,V-PEEK(3)*256
50030 SYS LOC,LX(A)*8,LY(A),LOC(A)
50040 FORB=1TORT:NEXT:NEXT:RETURN
50050 :
50060 REM ********************************
50070 :
60000 REM ROUTINES DI INIZIALIZZAZIONE
60010 POKE56,PEEK(46)+64:CLR:REM VENGONO LASCIATI 64*256 BYTES PER LE VARIABILI
60020 NFILE=100:NFOT=300
60030 DIM FL$(NFILE),NFOT(NFOT),LOC(NFILE),LX(NFILE),LY(NFILE)
60040 LIM=40700:LOC=40710:GOSUB60220:REM CONSIGLIATO PER GWBASIC LOC=820
60050 W=0
60060 W=W+1:IFW1>NFILE THEN PRINT"PER ALTRI FILES AUMENTARE NFILE":END
60070 PRINT"FILE NAME N."W" LOADING";:INPUTFL$(W):IFFL$(W)=""THEN60090
60080 GOTO60060
60090 PRINTCHR$(147):W=W-1:LOC(1)=(PEEK(56)+1)*256
60100 Q2=0:Q=0:FORQ1=1TOW:FL$=FL$(Q1):Q=Q+1:GOSUB60140:NEXT:W=W-Q2:GOTO20
60110 GOTO20:REM RITORNA AL PROGRAMMA
60130 :
60140 PRINT"FILE NAME:"FL$" RAM FREE:"LIM-LOC(Q):PRINTTAB(7)W-Q"FILES REMAINING"
60150 OPEN1,8,15:OPEN2,8,2,FL$+",S,R":INPUT#1,A,A$
60160 IFA<>0THENPRINTA$:Q=Q-1:Q2=Q2+1:GOTO60210
60170 INPUT#2,LU,LX(Q),LY(Q):LOC(Q+1)=LOC(Q)+LU
60180 IFLOC(Q+1)>LIMTHENPRINT"MEMORY FINISHED: NEXT FILES NOT LOADED":Q=Q-1:GOTO
60210
60190 FORA=0TOLU:PRINTCHR$(145)LU-A:GET#2,A$:Z=ASC(A$+CHR$(0))
60200 POKELOC(Q)+A,Z:NEXT
60210 CLOSE2:CLOSE1:RETURN
60220 PRINTCHR$(147)CHR$(17)"MOVIE MAKER (C) V. SPATARO: WAIT"LOC+200
60230 FORA=LOCTOLOC+200:PRINTCHR$(19)A:READB:POKEA,B:C=C+B:NEXT
60240 READB:IFC<>BTHENPRINT"ERROR IN DATA":END
60250 PRINTCHR$(147):RETURN
60260 DATA 032,241,183,134,007,032,241,183,134,008
60270 DATA 032,253,174,032,138,173,032,247,183,165
60280 DATA 020,133,004,165,021,133,005,169,254,045
60290 DATA 014,220,141,014,220,169,253,037,001,133
60300 DATA 001,165,008,133,244,169,000,133,243,160
60310 DATA 000,165,243,201,051,240,050,162,000,161
60320 DATA 004,201,000,240,089,234,234,145,002,230:REM 234,234 ANCHE 23,2
60330 DATA 004,165,004,201,000,208,002,230,005,200
60340 DATA 196,007,208,223,230,003,024,169,064,101
60350 DATA 002,133,002,144,002,230,003,198,244,165
60360 DATA 244,201,000,208,200,240,079,198,009,165
60370 DATA 009,201,000,208,002,198,010,165,009,201
60380 DATA 255,208,024,165,010,201,255,208,018,169
60390 DATA 000,133,243,230,004,165,004,201,000,208
60400 DATA 002,230,005,136,024,144,178,169,000,145,002:REM 145,2 ANCHE 234,234
60410 DATA 024,144,181,169,051,133,243,230,004
60420 DATA 165,004,201,000,208,002,230,005,132,006
60430 DATA 160,000,177,004,133,009,200,177,004,133
60440 DATA 010,164,006,024,144,177,169,002,005,001
60450 DATA 133,001,169,001,013,014,220,141,014,220
60460 DATA 096,23186:REM CHECKSUM ANCHE 23064
```

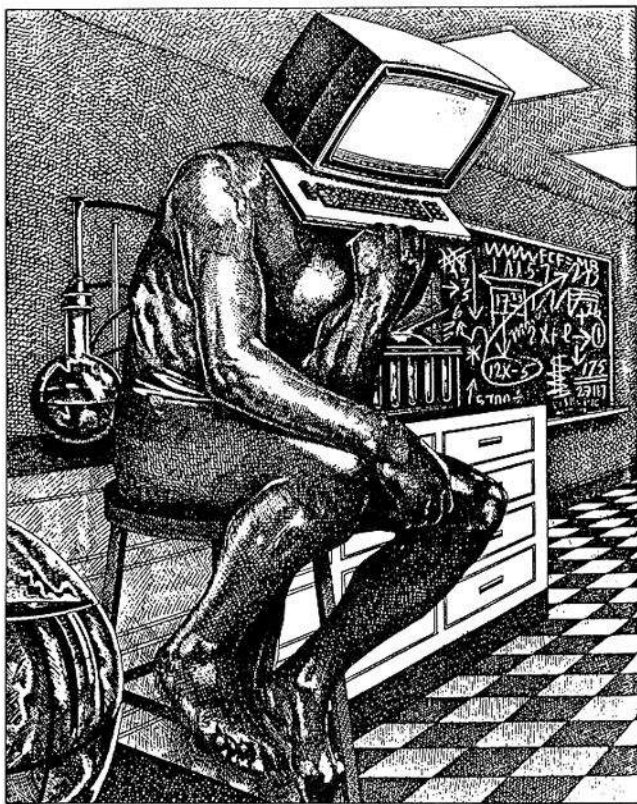
La tecnica di riproduzione del filmato è indipendente dal package grafico utilizzato

- Visualizza fotogrammi
- Interamente rilocabile
- Utilizzabile con qualsiasi routine grafica

```
0334 jsr $b7f1      ;prende lx*8
0337 stx $07        ;e lo mette in 07
0339 jsr $b7f1      ;prende ly
033c stx $08        ;e lo mette in 08
033e jsr $aefd      ;prende loc di partenza
0341 jsr $ad8a      ;della ram dati
0344 jsr $b7f7      ;
0347 lda $14        ;e li mette in
0349 sta $04        ;04
034b lda $15        ;e
034d sta $05        ;05
034f lda #$fe       ;toglie interrupt
0351 and $dc0e      ;
0354 sta $dc0e      ;
0357 lda #$fd       ;toglie la rom
0359 and $01        ;del kernal
035b sta $01        ;
035d lda $08        ;mette in f4 il numero
035f sta $f4        ;di righe
0361 lda #$00       ;azzera f3
0363 sta $f3        ;
0365 ldy #$00       ;e y
0367 lda $f3        ;e' in contemporanea
0369 cmp #$33       ;una serie di zeri ?
036b beq $039f      ;se si' salta
036d ldx #$00       ;se no azzera x
036f lda ($04),x    ;prende un dato da ram
0371 cmp #$00       ;e' un byte = 0 ?
0373 beq $03ce      ;se si' inizia serie 0
0375 nop            ; sostituibili con
0376 nop            ;ora ($02),y per merge
0377 sta ($02),y    ;e mette in bit map
0379 inc $04        ;incrementa puntatore
037b lda $04        ;alla ram dati
037d cmp #$00       ;lb e nel caso hb
037f bne $0383      ;
0381 inc $05        ;
0383 iny            ;incremento n. righe
0384 cpy $07        ;e' finito ?
0386 bne $0367      ;se no continua
0388 inc $03        ;se si somma 320 a pun-
038a clc            ;tatore bitmap
038b lda #$40       ;per passare alla
038d adc $02        ;linea successiva
038f sta $02        ;
0391 bcc $0395      ;
0393 inc $03        ;
0395 dec $f4        ;e decrementa n. linee
0397 lda $f4        ;
0399 cmp #$00       ;sono finite ?
039b bne $0365      ;se no continua
039d beq $03ee      ;se si riabilita
```

decremento contatore numero bytes - 0 da mettere in bit map

```
039f dec $09        ;decrementa n. bytes-0
03a1 lda $09        ;rimanenti
03a3 cmp #$00       ;
03a5 bne $03a9      ;se necessario anche hb
03a7 dec $0a        ;
03a9 lda $09        ;e' finito il n. di 0?
03ab cmp #$ff       ;(parte lb)
03ad bne $03c7      ;se no continua
03af lda $0a        ;e' finito il n. di 0?
03b1 cmp #$ff       ;(parte hb)
03b3 bne $03c7      ;se no continua
03b5 lda #$00       ;se si azzera f3 (fine
03b7 sta $f3        ;serie di 0) e aumenta
03b9 inc $04        ;puntatore a ram dati
03bb lda $04        ;
03bd cmp #$00       ;
03bf bne $03c3      ;
03c1 inc $05        ;e salta a altro incre-
03c3 dey            ;mento di puntatori
03c4 clc            ;
03c5 bcc $0379      ;
03c7 lda #$00       ;mette un byte-0
03c9 sta ($02),y    ;in bitmap
03cb clc            ;e continua
03cc bcc $0383      ;
```

procedura eseguita all7inizio di una serie di bytes-0

```
03ce lda #$33       ;imposta f3 per inizio
03d0 sta $f3        ;serie di 0
03d2 inc $04        ;e incrementa puntatori
03d4 lda $04        ;a ram dati
03d6 cmp #$00       ;lb e hb
03d8 bne $03dc      ;
03da inc $05        ;
03dc sty $06        ;salva numero righe
03de ldy #$00       ;azzera y
03e0 lda ($04),y    ;mette in 09 lb n.0
03e2 sta $09        ;
03e4 iny            ;
03e5 lda ($04),y    ;mette in 0a hb n.0
03e7 sta $0a        ;
03e9 ldy $06        ;
03eb clc            ;e salta
03ec bcc $039f      ;
03ee lda #$02       ;rimette rom
03f0 ora $01        ;
03f2 sta $01        ;
03f4 lda #$01       ;rimette l'interrupt
03f6 ora $dc0e      ;
03f9 sta $dc0e      ;
03fc rts            ;torna a basic
```

TUTTI I NUMERI DEL VOSTRO COMMODORE 64

Ovvero: come sommare, sottrarre, moltiplicare, dividere ed elevare a potenza ricorrendo al linguaggio macchina

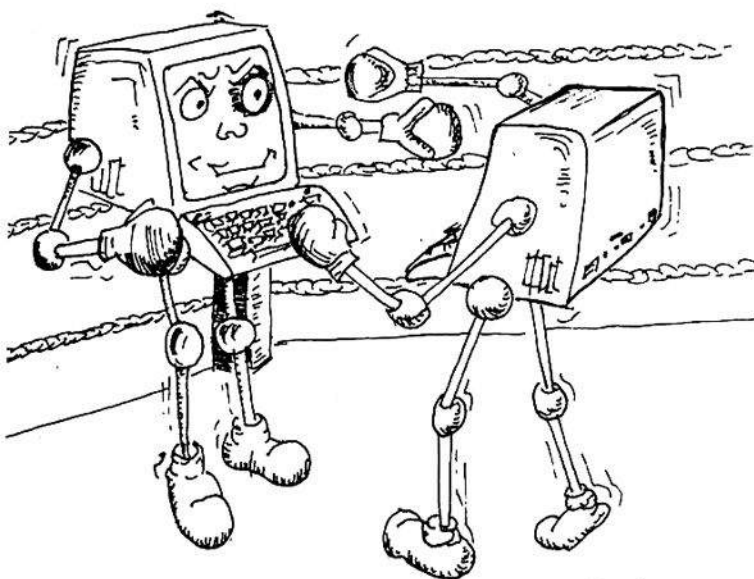
di **Giancarlo Mariani**

Sul n.43 di CCC (lontano giugno '87) comparve un articolo che spiegava il modo di adoperare, con il solo l.m. le funzioni matematiche del tipo Sin, Cos, ecc.

Con questo articolo completiamo il discorso, dal momento che mancavano all'appello le quattro operazioni più comuni (somma, sottrazione, moltiplicazione, divisione) e l'elevamento a potenza.

Siccome queste operazioni necessitano di due argomenti, eseguirle in l.m. è leggermente più complicato che non attivare le funzioni ad un solo argomento; in linea generale, tuttavia, il procedimento resta sempre lo stesso: lettura degli argomenti necessari e solito salto alle routines del S.O.

Per iniziare possiamo servirci del Basic come "mezzo" per passare i parametri alle routines l.m. dato che questo è il metodo più semplice; più avanti vedremo anche come renderci completamente indipendenti dal Basic, lavorando esclusivamente in l.m.

E' possibile utilizzare facilmente numerose routine contenute nelle Rom del C/64

Un semplice programma di Monitor sarà di aiuto per sperimentare le brevi routine proposte

Il primo passo

La prima operazione da esaminare è l'addizione. Tralasciando, per ovvi motivi, la spiegazione vera e propria dell'operazione, vediamo come il S.O. del C/64 provvede ad addizionare due numeri.

Innanzitutto va detto che tratteremo valori in virgola mobile che, come certamente saprete, sono memorizzati in una sequenza di 5 bytes, la cui descrizione particolareggiata sarebbe troppo lunga e complessa da spiegare e, in ogni caso, non utile al nostro scopo.

Nel C/64 vi sono due aree di 5 bytes ciascuna previste proprio per contenere due numeri in virgola mobile. Tali aree sono denominate "FLoating Point ACCumulator" (accumulatore in virgola mobile) n.1 e n.2. Di solito vengono abbreviate come FLP ACC#1 e FLP ACC#2, ma noi le chiameremo semplicemente ACC1 e ACC2.

Per eseguire un'addizione, bisogna semplicemente memorizzare i due operandi in ACC1 e ACC2, quindi saltare al "segmento" di S.O. che esegue l'addizione, ed infine leggere il risultato.

La routine di addizione del S.O. è posta a partire dall'indirizzo esadecimale $B86F. Questa legge i due argomenti da ACC1 e ACC2, li addiziona e quindi deposita il risultato in ACC1.

Dalla teoria alla pratica

Caricate un qualsiasi Tool che consenta di lavorare in linguaggio macchina (perfino un semplice Monitor va bene allo scopo) e battete, a partire da un indirizzo qualsiasi (ad es. $2000=8192), il breve programma di figura 1.

Tornate al Basic e digitate...

SYS 8192, N1, N2

...in cui N1 e N2, ovviamente, sono i due operandi da sommare, il cui valore può spaziare nell'intero campo dei numeri reali consentiti.

Una volta impartita la SYS, sullo schermo verrà stampato un numero, che sarà poi il risultato dell'addizione tra i due valori impostati.

Vediamo ora il funzionamento della routine:

Il primo JSR ($AEFD) serve solo per controllare che sia presente una virgola dopo la SYS. La routine $AD8A, richiamata subito dopo, cattura il numero reale presente dopo la virgola e lo trasferisce in ACC1.

A questo punto, prima di leggere un altro numero, dovremo provvedere a salvare quello appena letto, ed a questo ci pensa la routine $BBCA, che trasferisce ACC1 nell'area di memoria numerata da $57 a $5B (5 bytes).

I successivi due salti (JSR $AEFD e $AD8A) servono a "saltare" l'altra virgola e a prendere il secondo parametro, che sarà quindi posto in ACC1.

```
$2000:
JSR $AEFD ;Controlla la virgola.
JSR $AD8A ;Prende numero reale
           presente "dopo" virgola
JSR $BBCA ;Trasf. ACC1 da $57 a $5B.
JSR $AEFD ;Contr. virgola.
JSR $AD8A ;Prende altro num. reale.
LDA #$57  ;Trasferisce l' area da
LDY #$00  ;$57 a $5B
JSR $BA8C ;in ACC2.
JSR $B86F ;ADDIZIONE
           (ACC1 = ACC2 + ACC1)
JMP $AABC ;Stampa il valore di ACC1.
```

Figura 1

A questo punto delle operazioni abbiamo un operando in ACC1 e l'altro a partire da $57; siccome, però, dobbiamo avere i due operandi in ACC1 e ACC2, dovremo spostare il contenuto di $57-$5B in ACC2, operazione svolta da JSR $BA8C.

E' da notare che quest'ultima routine ha bisogno dell'indirizzo dal quale deve prelevare i 5 bytes, che gli viene fornito tramite i registri "A" ed "Y" del 6502: in Acc deve essere posta la parte bassa dell'indirizzo, mentre in Y la parte alta. Nel nostro caso (l'indirizzo è $0057) la parte bassa è $57 mentre quella alta è $00 (vedasi nel disassemblato LDA#$57 e LDY#$00).

Solo ora, finalmente, si può saltare alla routine che addiziona, posta all'indirizzo $B86F, che preleva il contenuto di ACC2, lo somma con ACC1 e quindi ne deposita il risultato in ACC1, che potrà, infine, essere visualizzato mediante la routine $AABC.

Vi consigliamo di effettuare numerose prove attivando la routine proposta, tenendo presente che i due operandi possono essere numeri reali qualsiasi (compresi tra -1E37 e 1E37).

Avete dunque imparato ad eseguire una semplice operazione in l.m. ma per operare solo con questo linguaggio è necessario apportare una modifica alla routine appena vista.

I numeri in virgola mobile (ricordiamo che sono formati da una sequenza di 5 bytes) sono piuttosto duri da trattare direttamente in l.m. Volendo inserire, in 5 bytes di memoria, un valore in virgola mobile senza servirci del Basic come tramite, dovremmo eseguire calcoli talmente complessi da farci passare la voglia di usare questo metodo.

In l.m. i numeri più facili da trattare sono, infatti, quelli interi compresi tra 0 e 65535, che possono essere rappresentati in memoria con 2 bytes, di solito contigui: il byte alto (HI) ed il basso (LO). Chiamando N il numero da convertire, ricordiamo che:

```
HI = INT(N/256)
LO = N-HI*256
```

Lo svantaggio nel trattare questi numeri è che il campo si restringe tra 0 e 65535 e si perdono i decimali, che però, lavorando in l.m, servono a poco.

Siccome abbiamo detto che la routine di somma accetta in ingresso due numeri in virgola mobile, dovremo convertire il valore intero in ingresso in un numero FLP. Supponendo di avere il numero intero nei due bytes $FB (251) e $FC (252), per convertirlo in FLP esiste una routine del S.O. posta da $B391 che deve avere in ingresso il numero nei registri Y (parte bassa) e A (parte alta); effettuata la conversione, la routine deposita il risultato in ACC1.

In altre parole:

```
LDY $FB; Carica in Y la parte bassa
LDA $FC; e in A l'alta del numero.
JSR $B391; Lo converte in FLP e lo mette in
ACC1
```

Utilizziamo ora le nozioni apprese: scomponiamo il primo operando in byte basso / byte alto e poniamolo nei due bytes $FB e $FC, anche per mezzo di una banale...

```
POKE 251,LO: POKE 252,HI
```

Allo stesso modo poniamo il secondo operando nei bytes $FD e $FE (253 e 254) e diamo il tradizionale SYS 8192 alla routine di figura 2.

Il suo funzionamento è simile a quello della routine precedente: invece di prendere i numeri dal Basic, li carica da 4 locazioni di memoria ($FB, $FC per il primo; $FD, $FE per il secondo) e, tramite la routine $B391, li converte in numeri in virgola mobile (FLP).

Siamo dunque in grado di gestire una routine che accetta due numeri interi (e quindi trattabili facilmente da l.m.) in ingresso, ma fornisce ancora un risultato in virgola mobile, non facilmente sfruttabile in l.m.

La cosa più semplice che viene in mente di fare è quella di convertire anche il risultato in numero intero, anche perchè, avendo

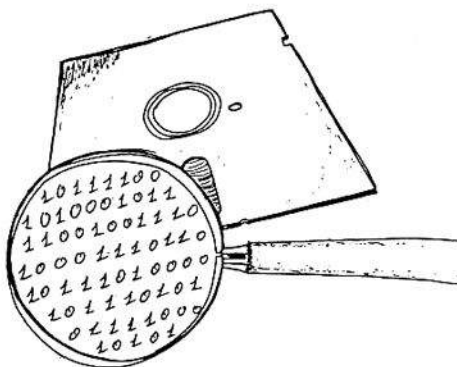
Alcune funzioni richiedono il preventivo "caricamento" dell' accumulatore e del registro Y

```
$2000:
LDY $FB     ;Carica in Y la parte
             bassa del valore
LDA $FC     ;e in A la parte alta del
             primo numero.
JSR $B391   ;Converte in FLP e deposita
             in ACC1
JSR $BBCA   ;Trasf. ACC1 in $57-$5B.
LDY $FD     ;Carica in Y la parte bassa
LDA $FE     ;e in A la perte alta del
             secondo numero.
JSR $B391   ;Converte in FLP e mette
             in ACC1
LDA #$57    ;Trasferisce i 5 bytes
LDY #$00    ;da $57 a $5B
JSR $BA8C   ;in ACC2.
JSR $B86F   ;Esegue l'addizione.
JMP $AABC   ;Stampa il risultato.
```

Figura 2

In l.m. capita raramente di operare in Floating Point perchè si ha a che fare quasi sempre con valori interi

due numeri interi in ingresso, un risultato in virgola mobile è perfettamente inutile. La routine che provvede a svolgere il compito richiesto è posta nel S.O. all'indirizzo $B7F7.

Questa preleva il contenuto di ACC1 (in cui, lo ricordiamo, era presente il risultato dell'addizione), lo converte in intero a 2 bytes e lo memorizza in $14 (parte bassa) e $15 (parte alta).

Riprendete ora l'ultima routine digitata e, tralasciando l'ultimo JMP ($AABC) aggiungete il segmento di l.m. riportato in figura 3.

Il JSR$B7F7, come detto, serve a convertire il risultato in numero intero ed a trasferirlo nelle locazioni $14/$15. La routine $BDCD stampa il numero contenuto in X/A ed equivale all'istruzione Basic:

PRINT X+(A*256)

Quanto detto sinora per l'addizione, vale anche per le altre operazioni matematiche, quali:

```
JSR $B7F7   ;Converte ACC1 in intero
             e lo pone in $14 / $15.
LDX $14     ;Carica il byte basso in X.
LDA $15     ;e il byte alto in A.
JMP $BDCD   ;Stampa l'intero in X/A.
```

Figura 3

Sottrazione: $B853
Moltiplicazione: $BA2B
Divisione: $BB12
Elevamento a potenza: $BF7B

Per utilizzare queste ultime routines, sarà sufficiente sostituire, nel programma visto prima, il JSR $B86F (che esegue l'addizione) con uno qualsiasi di quelli sopra elencati. I ragionamenti fatti per l'addizione sono identici in tutti gli altri casi. Ovviamente nell'operazione di elevamento a potenza il primo parametro sarà la base mentre il secondo l'esponente.

Una curiosità

Il lettore più attento avrà notato che nella routine i due operandi vengono caricati in modo inverso, ossia il primo va a finire in ACC2, mentre il secondo in ACC1.

Questo perchè le routines del S.O. funzionano in modo inverso, ossia considerano come primo parametro quello contenuto in ACC2, mentre come secondo quello di ACC1. Ovviamente tutti noi sappiamo che per sommare o moltiplicare non è necessario rispettare un ordine negli operandi, cioè...

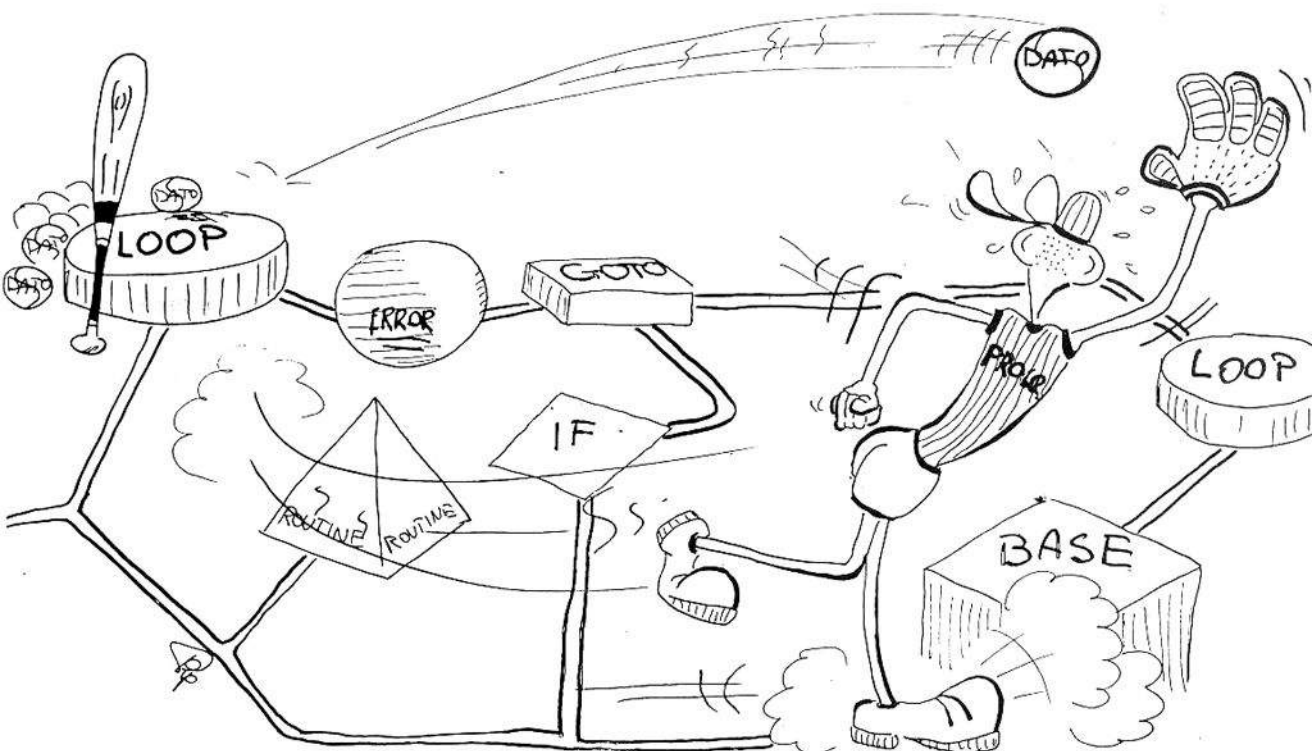
$A + B = B + A$
$A \times B = B \times A$

Ciò non è vero nel caso di sottrazioni, divisioni ed elevamenti a potenza, ed è questa la ragione per cui vengono tra loro scambiati gli operandi. Un'ultima nota prima di concludere: la routine di conversione da intero a FLP ($B391) accetta numeri interi compresi di segno. Ciò significa che quelli maggiori di $7FFF (cioè con byte alto maggiore o eguale al valore $80) vengono considerati negativi: tenetene conto nello svolgimento delle operazioni.

La sintassi, in tutti i casi, resta sempre la stessa:

SYS XXXX, op1, op2

in cui XXXX è l'indirizzo di partenza che NON è quello delle routines S.O. elencate sopra, ma quello dell'intera routine scritta dall'utente; negli esempi trattati l'indirizzo è $2000 (=8192).

Nel caso dell'elevamento a potenza op1 è la base mentre op2 è l'esponente.

INVITO ALLA KERMESSE DEL KERNEL DI AMIGA

Partecipate anche voi alla grande festa che si verifica nei pressi del cervello di Amiga, ma attenti a non farvi coinvolgere troppo: potreste scottarvi!

di **Luigi Callegari**

Che cosa è il Kernel

Il "Kernel" è il programma che giace nei 192K di ROM (memoria a sola lettura) di Amiga e che lo trasformano da un mucchio di stupido silicio nell'apparecchio dalle prodigiose prestazioni che tutti conosciamo.

In poche altre macchine il software e lo hardware si trovano a lavorare così vicino come in Amiga. Chi conosce la storia avventurosa della sua creazione, troverà la spiegazione di ciò nel fatto che i tecnici hardware ed i programmatori hanno lavorato sempre fianco a fianco, influenzandosi di continuo.

Si pensi, ad esempio, che le caratteristiche esclusive del Blitter (programmabile!) di Amiga sono state suggerite, dai programmatori del sistema operativo, ai tecnici costruttori; Mical e Luke, da parte loro, faticarono non poco per realizzare le interfacce software necessarie a sfruttare a fondo i super circuiti creati da Jay Miner (e soci).

Lo hardware ed il software di Amiga cooperano, come i loro "genitori", per ottenere le migliori prestazioni. Ad esempio, il Copper (coprocessore video) è in grado di sincronizzarsi con l'immagine tracciata dal tubo catodico, senza disturbare il processore 68000, od il suo

La peculiarità di Amiga è la stretta connessione tra l'hardware ed il software, che agiscono l'uno per l'altro

"Dentro" l'Amiga

Chi si avvicina alla programmazione evoluta di Amiga, in linguaggio C oppure Assembly, incontra invariabilmente difficoltà enormi.

Innanzitutto il sistema operativo (S.O.) è notevolmente più sofisticato e complesso di quelli residenti nelle vecchie macchine dotate di microprocessori ad 8 bit: per convincersene è sufficiente paragonare i ben 192K della ROM di Amiga con i 16K tipici di un C/64.

Inoltre le novità del multitasking, la grafica di tipo "Bit-Plane" (invece della tipica "Bit-Mapped"), le esclusività delle directory ad albero di AmigaDOS, e tante altre sofisticazioni, partecipano attivamente a confondere le idee anche a chi ha programmato, poco o tanto, in Assembly su vecchi computer, che si trova nelle condizioni, praticamente, di studiare tutto daccapo.

Inoltre, e questo crediamo sia il fatto più importante, la documentazione sulle macchine di questo tipo si divide in due grandi categorie.

Alla prima appartengono i volumi destinati ai principianti "assoluti"; ci riferiamo ai libri editi in Italia in "anteprima" da una nota casa editrice la quale, chissà perchè, si ostina a divulgare volumi che sono quasi sempre mere traduzioni maccheroniche del manuale, condensate in modo sadicamente criptico e riempite accuratamente di errori prima di essere vendute a cifre tradizionalmente esorbitanti.

Alla seconda categoria appartiene la documentazione originale in lingua inglese, composta da ben quattro volum(on)i enciclopedici di centinaia di pagine ciascuno, validissimi ma assolutamente illeggibili per chi non ha un quoziente intellettivo tipo Einstein, come sottolineano gli stessi programmatori professionisti americani abituati a lavorare con mainframe ed a sviluppare progetti di intelligenza artificiale sulle pagine di riviste lette in tutto il mondo.

Dunque non possiamo certo pretendere di insegnare a programmare, in Assembly o in "C", un computer del calibro di Amiga, almeno non nell'esiguo spazio concesso dalla nostra pur pingue rivista.

Crediamo però che molti troveranno utile sapere, pur se per grandi linee, che cosa c'è nelle ROM di Amiga, in modo che, trovato il coraggio di comprare qualche libro sull'argomento (ma che sia originale in inglese, per carità), possano destreggiarsi almeno nella consultazione dell'indice del volume.

Ricordiamo nuovamente, a scanso di equivoci, che gli autori dei libri americani invariabilmente danno per scontato che il lettore abbia almeno tre lauree nel settore informatico.

Tenteremo disperatamente di fornire un riassunto delle "funzioni" disponibili ai programmatori C ed assembly, lasciando la descrizione delle lunghissime "strutture" di parametri alla documentazione ufficiale.

bus, nonchè caricare direttamente alcuni valori all'interno dei registri della CPU o richiedere direttamente un interrupt, che il modulo software Exec poi tratta in multitasking.

La memoria RAM di Amiga funziona ad una frequenza di clock doppia (14 MHz) rispetto alla CPU (7.15909 MHz), cosicchè i circuiti DMA (memoria ad accesso diretto) che debbono accedervi, possono farlo nei cicli in cui il 68000 sta facendo i suoi calcoli interni. Tutti i trasferimenti di dati avvengono tramite canali DMA senza disturbare il 68000 il quale, quando sta lavorando in zone di memoria poste al di fuori dei primi 512K (area gestibile in comune dai vari coprocessori) non viene mai rallentato da alcun fenomeno, ed opera, pertanto, alla massima velocità.

Altro esempio di perfetta simbiosi hard/soft si ha nella gestione delle periferiche (stampanti, modem, floppy...), tutte "servite" da interruzione. In questo modo nessuna periferica richiede il monitoraggio da parte del 68000 (la vetusta tecnica chiamata "polling interrupt") sinchè non debbono essere trasferiti dati. La procedura "Exec", come infatti vedremo, provvede ad un completo meccanismo di trattamento delle richieste di interruzione da periferiche, secondo criteri di priorità ottimizzati.

Exec, cuore del Kernel

Il cuore del sistema è EXEC (contrazione di "Executive", esecutivo), che consiste in una collezione di routines ROM poste alla base del software in ROM (Intuition, AmigaDOS...).

I suoi creatori lo hanno ottimizzato in termini di spazio, prestazioni, chiarezza di uso, creazione e gestione di "liste".

Proprio la "lista" è il componente principale, il mattone modulare di Exec. Tutto gira attorno a questo concetto che, una volta appreso fino in fondo, permette l'uso delle più raffinate funzioni di Exec, senza problemi.

Exec è il punto di partenza di tutti gli altri programmi posti in ROM, anche perchè controlla tutte le interruzioni ed i "task" (compiti). Ogni componente di Exec è stato concepito per lavorare il più a lungo possibile da solo.

Principi di software nel Kernel

Exec, dunque, gestisce più task contemporaneamente, suddividendo in modo adeguato il lavoro della CPU 68000. Ciascun task può essere concepito dal programmatore come se disponesse per intero della CPU, nella maggior parte delle applicazioni.

Qualunque "pezzo" di software operante al di fuori di Exec è da concepirsi come un task.

Un "device", in ambiente Kernel, è uno speciale meccanismo di I/O (immissione ed emissione di dati) che usa particolari task per creare accessi sistematici alle risorse del sistema.

Una "libreria" è una collezione di routines assembly che risiedono in ROM, oppure che sono state caricate da disco. Le routines di libreria vengono richiamate indirettamente; ne consegue che il sistema (o gli applicativi che le usano) non debbono necessariamente conoscerne l'indirizzo di accesso. A parte un solo indirizzo, quello dell'accesso all'archivio Exec, non vi sono indirizzi fissi ed assoluti in Amiga, quindi non se ne richiede mai l'uso.

Usando, ad esempio, la libreria grafica di supporto (anche da Basic) memorizzata su disco, è possibile sia effettuare operazioni a livello molto basso (gestire direttamente il Copper od il Blitter tramite i loro registri hardware) sia molto complesse (come riempire poligoni).

Anche "Intuition" è una libreria, che usa a sua volta la libreria grafica per provvedere ad una interfaccia grafica semplificata (Workbench) con l'utente.

Multitasking

E' bene, ora, chiarire che cosa indichi esattamente il termine "multitasking", del quale molti riempiono la bocca (e le pagine di riviste) senz sapere che cosa sia, magari dando l'idea che si tratta di un tipo di lavoro ad interrupt, vecchio stile.

Nel sistema multiprogrammabile di Amiga ogni unità esecutiva (task), eccetto quella svolta dal coprocessore, dispone di un proprio completo ambiente di lavoro, in pratica un'intera macchina. Infatti ogni task può disporre dei suoi registri, del suo stack, del suo stato di processo e può accedere alle periferiche di I/O esattamente come se fosse l'unico task in esecuzione nel sistema in quel momento, senza preoccuparsi minimamente di eventuali altri task che competono per le risorse comuni. Cio non è vero solo per programmi molto particolari, che debbano lavorare in situazioni di tempo critico.

Ben applicati, i task sono molto semplici e lineari da usare: programmi, periferiche e lo stesso AmigaDOS sono stati costruiti su di loro.

Exec è la "forza" prima" su cui si basa l'intero sistema Amiga: è il cuore del cuore del computer

Una parola di 32 bit sovrintende alle vitali informazioni per la gestione multitasking di Amiga

Il programmatore, ad ogni modo, può permettersi di ignorare le possibilità di multitasking di Amiga e lavorare come se si trattasse di una macchina "normale"; eccettuate, ovviamente, eventuali applicazioni molto critiche e speciali.

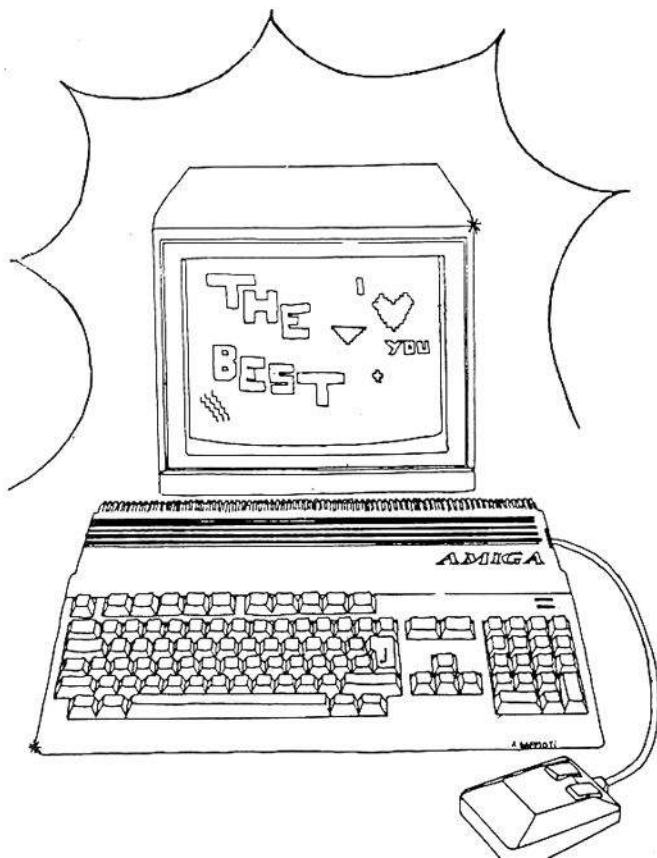
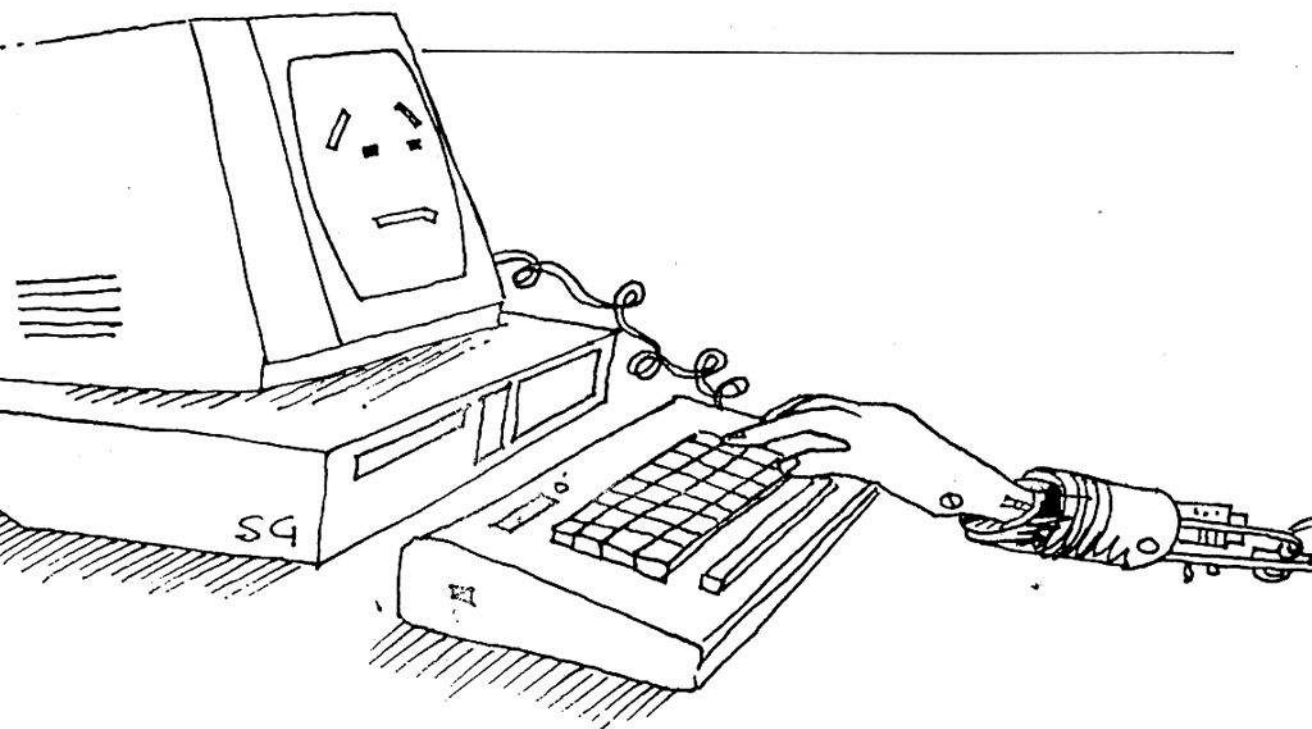
L'unico accorgimento da seguire, nella stesura di un programma che operi in ambiente multiprogrammabile, consiste nella necessità di restituire esplicitamente al sistema il processore e le sue risorse quando non necessarie, onde permettere agli altri task di lavorare felicemente.

Per esempio, quando il nostro task deve aspettare una interruzione da qualche periferica, si può chiamare la funzione di Exec "Wait()" che provvede automaticamente ad eseguire altri task eventualmente in stato di attesa e di ritornare al task primitivo solo quando (e se) giunge l'attesa richiesta di interrupt dalla periferica.

Dal momento che Amiga dispone di un solo microprocessore, i vari task vengono da questo serviti (cioè, eseguiti) in intervalli di tempo molto brevi ed, ovviamente, uno solo per volta. Ogni task ha un numero di priorità (modificabile perfino da CLI con il comando "Changetaskpri"), che indica praticamente "quanto" è importante la sua esecuzione. I parametri variano da -128 a +127). Per default viene assegnato solitamente il valore zero. Un task con priorità più elevata viene eseguito tutte le volte che segnala di essere pronto, anche se ciò richiede l'interruzione di un task a priorità più bassa. Se due o più task hanno la medesima priorità, essi vengono eseguiti alternativamente per 64 millisecondi ciascuno. Quando un task ha finito il suo lavoro deve chiamare la già nominata funzione di Exec chiamata Wait().

Messaggi e segnali

I task comunicano tra loro per mezzo di "messaggi" e di "segnali".

Il segnale è il sistema più semplice e rapido da trattare, consistendo, in pratica, in un solo bit all'interno di una parola lunga 32 bit (chiamata "parola bit di segnalazioni"). Ogni task ha quindi 32 bit di segnalazione, alcuni dei quali riservati al sistema. L'argomento della funzione Wait() è, appunto, una parola di 32 bit in cui i bit attivi indicano i segnali che il task attende. In

questo modo, quando il task chiama la funzione Wait() esso informa direttamente Exec sui vari "eventi" che sta attendendo, in modo da essere riattivato automaticamente non appena si verificano.

La funzione AllocSignal() di Exec permette ad un task di allocare i bit di segnalazione delle informazioni da trasmettere. Inoltre un task può rendere il segnale disponibile globalmente a tutti gli altri task, oppure come una informazione destinata ad un solo task.

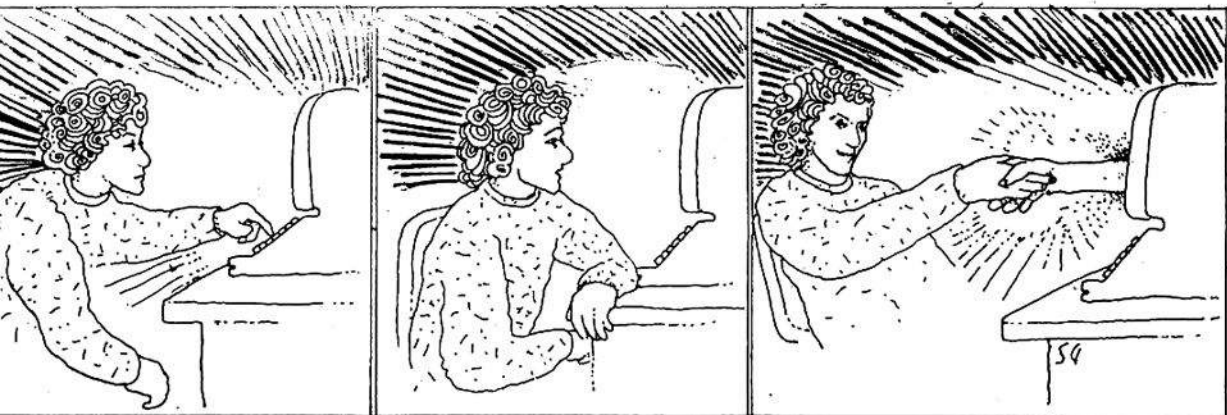
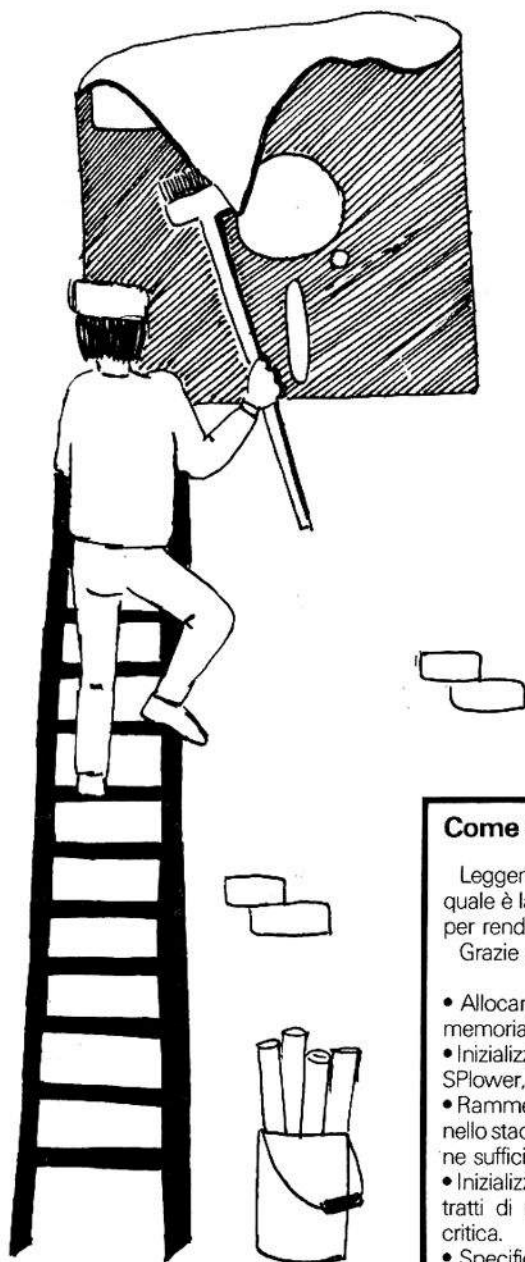
I task comunicano tra loro sia usando segnali che la funzione di Exec Signal().

Come abbiamo detto, oltre ai segnali esistono messaggi per comunicare tra task. In questo caso i diversi task possono leggere le informazioni circa le porte di intercomunicazione per mezzo di una variabile globale (accessibile da tutti) o leggendo il nome pre-assegnato delle porte.

Vi sono parecchie funzioni Exec per la gestione delle porte. La creazione avviene per mezzo della funzione CreatePort(), che permette anche di assegnare un nome alla porta stessa. Gli altri task possono reperire una porta di comunicazione usando la funzione Findport().

La funzione CreatePort(), in effetti, usa un'altra funzione dopo avere allocato ed inizializzato la memoria necessaria alla gestione della porta, e precisamente chiama AddPort() per installare effettivamente il nuovo messaggio nel sistema.

La gestione della RAM in un ambiente "delicato" come quello multitasking di Amiga avviene tramite numerose funzioni di Exec.

La memoria Ram viene utilizzata solo quando serve

Le funzioni basilari sono comunque AllocMem() e FreeMem(). La prima permette di ottenere la quantità di memoria richiesta dal sistema (se disponibile), eventualmente azzerata; la seconda, come intuitivo, rilascia una zona di memoria, rendendola nuovamente accessibile al sistema. Da notare che AllocMem() permette di specificare anche che tipo di memoria si desidera: se CHIP RAM (primi 512K, indispensabili per contenere routine per i coprocessori), oppure FAST RAM.

La funzione PutMsg() invia un messaggio ad una porta ed è possibile decidere di attendere una risposta attraverso la porta stessa (I/O sincrono) oppure proseguire l'elaborazione (I/O asincrono).

Un task riceve un messaggio da una porta usando la funzione GetMsg() che restituisce l'indirizzo del messaggio in attesa di elaborazione, oppure zero se non ve ne sono.

Allocazione della memoria

La gestione effettiva della memoria allocata da parte del task avviene comunque tramite le funzioni Allocate() e Deallocate().

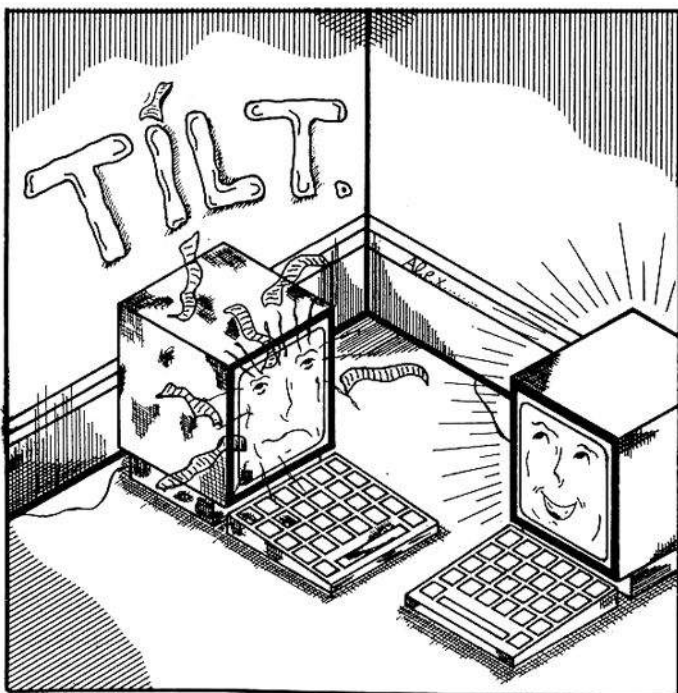
Inoltre, la memoria riservata per un task può essere automaticamente liberata alla fine della sua esecuzione, usando la funzione Alloc Entry().

Importanti sono anche le funzioni AllocRemember() e FreeRemember().

Come aprire un task

Leggendo la documentazione americana ufficiale, risulta arduo capire quale è la sequenza di operazioni necessarie per aprire un task, ovvero per rendere effettivamente operativo un programma assembly o C.

Grazie a C.C.C. (modestia a parte) ecco la preziosa successione:

- Allocare spazio per lo stack del task (vedere funzioni di allocazione memoria).
- Inizializzare le variabili di sistema per la gestione dello stack: SPPUpper, SPlower, SPRegister.
- Rammentare che tutte le chiamate di funzioni necessitano di memoria nello stack e che il programmatore è tenuto ad assicurare una dimensione sufficiente. La dimensione mimina dello stack è di 66 bytes.
- Inizializzare opzionalmente la priorità del Task. E' opportuno nel caso si tratti di un task che deve lavorare in situazioni di temporizzazione critica.
- Specificare il nome del task. Anche se non è obbligatorio, è ottima regola.
- Creare una porta di comunicazione usando CreatePort(), se serve.
- Chiamare AddTask() per indicare al sisistema che, da questo momento, è presente un nuovo task da eseguire.

SE LA MEMORIA NON VACILLA...

Che cosa si intende dire con l'espressione "selezionare il banco" riferito alla memoria del C/64

di **Giancarlo Mariani**

Quando si accende il C/64, non si può fare a meno di notare i messaggi che compaiono sul video, ed in particolare i due che informano sulla quantità di memoria Ram del sistema (64K) e su quella effettivamente disponibile per i programmi Basic (38911 bytes).

Sembrerebbe che la Ram promessa non sia però presente nel computer: invece c'è, ma non si "vede".

Nel C/64 c'è una locazione (la n.1) che serve per decidere in che modo la memoria del computer deve essere ripartita. Normalmente questa locazione contiene il valore esadecimale $37 che, scomposto in binario, diventa 00110111. Di questi otto bit, a noi interessano solo i primi 3 (partendo da destra), ossia i bit n.0, 1 e 2 che controllano, rispettivamente, varie funzioni:

0: Segnale LO-Ram
1: Segnale HI-Ram
2: Segnale CHAREN

Prima di spiegare a che cosa servono, osserviamo, in figura 1, la mappa di memoria del C/64 al momento dell'accensione.

Facendo un po' di conti si nota che, in to-

Una Rom ed una Ram "coabitano" negli stessi indirizzi, come due famiglie su uno stesso pianerottolo

Il computer può "vedere" un solo banco alla volta: una Rom oppure una Ram

```
Locazione 1:
Bit:  2 1 0
Mode: 1 1 1

0000-03FF: Pagina 0             (Ram)
0400-07F7: Mappa video          (Ram)
07F8-07FF: Locaz. per sprite    (Ram)
0800-9FFF: Programmi Basic      (Ram)
A000-BFFF: Interprete Basic     (Rom)
C000-CFFF: 4K liberi per l.m    (Ram)
D000-DFFF: Input / Output       (I/O)
E000-FFFF: Sistema Operativo    (Rom)
```

Figura 1

Configurazione della memoria del C/64 al momento dell'accensione. In questo caso i bit 0, 1, e 2 della locazione 1 sono settati (posti a 1)

```
Locazione 1:
Bit:  2 1 0
Mode: 1 1 0

0000-03FF: Pagina 0             (Ram)
0400-07F7: Mappa video          (Ram)
07F8-07FF: Locaz. per sprite    (Ram)
0800-9FFF: Programmi Basic(?)(Ram)
A000-BFFF: Area Ram             (Ram)
C000-CFFF: 4K liberi per l.m    (Ram)
D000-DFFF: Input / Output       (I/O)
E000-FFFF: Sistema operativo    (Rom)
```

Figura 2

Resettando il bit 0 la Rom del Basic "scompare" per far apparire, al suo posto, la Ram sottostante.

tale, la Ram presente ammonta a "soli" 44K, e che quindi siamo ancora lontani dai fatidici 64K promessi. L'area di Ram mancante è comunque presente nel C/64, ma si trova nascosta "sotto" le due Rom della macchina (interprete Basic e S.O.): proprio così, è posta esattamente nelle stesse locazioni.

Il modo per comunicare al C/64 il desiderio di avere disponibile la Ram piuttosto che la Rom (o viceversa) è quello di alterare la locazione 1 vista in precedenza. Ricordiamo che i tre bit considerati erano tutti posti a 1, il che vuol dire: Rom abilitate e generatore di caratteri escluso. Vediamo comunque in dettaglio il significato dei tre bit:

Bit 0 (lo-Ram): controlla la Ram posta sotto l'interprete Basic; se il bit è settato (posto a 1), sarà la Rom ad essere vista dal C/64, mentre se si pone a 0 si abiliteranno gli 8K di Ram posti sotto l'interprete escludendoli, quindi, dall'orizzonte della macchina. La mappa di memoria, in questo caso (bit 0 posto a 0), viene trasformata come in figura 2.

In questa possiamo notare che nelle locazioni numerate da A000 a BFFF al posto dell'interprete Basic troveranno posto 8K di Ram sfruttabili per programmi l.m. pagine grafiche o, addirittura, un altro linguaggio.

Bisogna però prestare molta attenzione nell'abbassare il bit 0, perchè dal momento che si disabilita l'interprete Basic, il C/64 non sarà più in grado di riconoscere i comandi che verranno impartiti, e quindi sarà possibile lavorare solo in l.m.

Il bit 1 della locazione 1 (hi-Ram) provvede a selezionare la funzione degli ultimi 8K di memoria del C/64: se il bit è posto a 1, questi 8K saranno di Rom (S.O.), mentre se si abbassa a 0 verrà abilitata la Ram (come in figura 3).

Questo bit è ancora più "delicato" del precedente, dato che il S.O. del C/64 contiene le numerose routines di interrupt: ne consegue che prima di abilitare la Ram sarà necessario disabilitare l'interrupt, altrimenti il computer, non trovando quello che ci dovrebbe essere, si inchioda.

La sequenza per leggere un dato posto nella Ram posta sotto il S.O. dovrebbe essere la seguente:

1 - Disabilitare l'interrupt
2 - Abilitare la Ram (porre il bit a 0)

```
Locazione 1:
Bit:   2 1 0
Mode:  1 0 1

0000-03FF: Pagina 0                (Ram)
0400-07F7: Mappa video             (Ram)
07F8-07FF: Locaz. per sprite       (Ram)
0800-9FFF: Programmi Basic         (Ram)
A000-BFFF: Interprete Basic        (Rom)
C000-CFFF: 4K liberi per l.m       (Ram)
D000-DFFF: Input / Output          (I/O)
E000-FFFF: Area Ram                (Ram)
```

Figura 3

Il bit 1 gestisce la Ram sottostante la Rom del Sistema Operativo (S.O.)

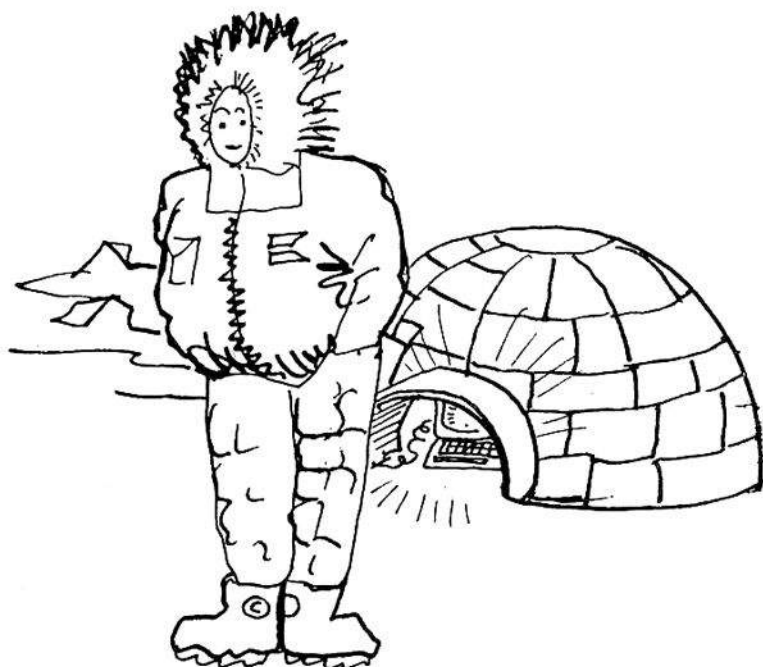
La locazione 1 è il mago che fa apparire (o scomparire) Rom e Ram

```
Locazione 1:
Bit:   2 1 0
Mode:  1 0 0

0000-03FF: Pagina 0                (Ram)
0400-07F7: Mappa video             (Ram)
07F8-07FF: Locaz. per sprite       (Ram)
0800-9FFF: Programmi Basic(?)      (Ram)
A000-BFFF: Area Ram                (Ram)
C000-CFFF: 4K liberi per l.m       (Ram)
D000-DFFF: Input / Output          (I/O)
E000-FFFF: Area Ram                (Ram)
```

Figura 4

Resettando i bit 0 ed 1 si disabilita sia la Rom del Basic che del S.O. In questo caso, però, al momento della "scomparsa" delle Rom è necessario che il computer "veda" routine l.m. che gli consentano di proseguire l'elaborazione.

L'interrupt gioca un ruolo fondamentale: guai a trascurarlo nella selezione dei banchi.

```
Locazione 1:
Bit:  2 1 0
Mode: 0 1 1

0000-03FF: Pagina 0             (Ram)
0400-07F7: Mappa video          (Ram)
07F8-07FF: Locaz. per sprite    (Ram)
0800-9FFF: Programmi Basic      (Ram)
A000-BFFF: Interprete Basic     (Rom)
C000-CFFF: 4K liberi per l.m    (Ram)
D000-DFFF: Generatore carat.    (Rom)
E000-FFFF: Sistema Operativo    (Rom)
```

Figura 5

Il bit 2 della locazione 1 gestisce il generatore dei caratteri visibili. Per farlo "apparire" è necessario "oscurare" il chip I/O presente, di solito, negli stessi indirizzi.

3 - Leggere il dato
4 - Disabilitare Ram (rimettere bit a 1)
5 - Abilitare l'interrupt

La sequenza, escludendo i passi 1 e 5, è valida anche per leggere un dato dagli 8K di Ram posti sotto il Basic, a patto, ovviamente, di non usare la funzione Peek del Basic, escluso dalla stessa operazione!

E' utile sapere che per scrivere un dato in queste Ram non è necessario rispettare una sequenza particolare, perchè le Ram, durante un'operazione di scrittura sono sempre abilitate; anche una semplice POKE eseguita da Basic può quindi servire allo scopo (una Peek, invece, si limiterebbe a leggere i dati contenuti nelle Rom).

Naturalmente le Ram nascoste possono essere abilitate insieme abbassando i bit 0 e 1 contemporaneamente. In questo caso la mappa diventa simile a quella di figura 4.

Un bit speciale

Il bit 3 della locazione 1 (charen) serve per uno scopo piuttosto particolare, ossia abilitare la lettura del generatore di caratteri, normalmente nascosto sotto gli indirizzi di I/O (D000-DFFF).

Il generatore consta di 4K di Rom e contiene le informazioni che necessitano al C/64 per visualizzare l'intero set di caratteri (lettere, numeri, simboli speciali, e così via).

Normalmente si usa leggere il contenuto del generatore per trasferirlo in altra zona Ram, ove modificare con calma i caratteri da programmare. La sequenza per leggere i dati contenuti nei 4K di Rom è la stessa vista prima, con la piccola differenza che può anche essere eseguita da Basic, ricordando che il comando...

Poke 56334, Peek(56334) And 254

...disabilita l'interrput e...

Poke 56334, Peek(56334) Or 1

...lo riabilita.

In figura 5 è mostrata la mappa di memoria contenente il generatore di caratteri al posto degli indirizzi di I/O.

WIZ2 - IMMAGINE MARSILII & TERZANI - CIELO DI RIETI
Il lettore di VR Videoregistrare è giovane, dinamico, creativo. Di cultura e reddito superiore alla media, possiede spesso più di un videoregistratore, oltre all'impianto hi-fi e al computer: nel tempo libero, non rinuncia a viaggi in Italia e all'estero, e a cinema, teatro e spettacoli sportivi in genere. Usa il videoregistratore non solo per i programmi tv o preincisi, ma anche per riprendere i momenti felici in famiglia, per creare una videoteca personale.
E tu, che lettore sei?
LEGGO VR PERCHÈ SOSTIENE I MIEI INTERESSI
VR
VIDEOREGISTRARE
L'immaginazione al potere

IN EDICOLA

SE POTESSI AVERE 1000 LIRE AL MESE...

Le tasse vanno pagate, questo è noto e doveroso. Vi invitiamo, quindi, a fare considerazioni di vario tipo su di un tema così scottante

di **Alessandro de Simone**

Due amici chiacchierano del più e del meno.

"Come mi piacerebbe -dice il primo- avere i soldi per acquistare un tansatlantico!".

"Ma sei matto? -ribatte l'altro- Che te ne faresti di un transatlantico?"

"Non ho detto che voglio un transatlatico -precisa il primo- ma solo che mi piacerebbe avere i soldi per acquistarlo..."

Le tasse

Il modello 740 di quest'anno presenta alcune modifiche, tra le altre novità, per ciò che riguarda gli scaglioni di reddito imponibili.

Forse non tutti i lettori conoscono il nostro sistema di tassazione, che è bene riassumere e considerare.

Tutti coloro che guadagnano (tecnicamente: che producono reddito)

SCHEDA TECNICA

Software applicativo per calcoli contabili

Hardware richiesto: qualsiasi computer Commodore

Anche il programma pubblicato in queste pagine è contenuto nel disco "Directory" di questo mese.

devono contribuire alle spese dello stato, secondo le proprie disponibilità.

Chi guadagna poco, in poche parole, paga poco; i più ricchi pagano di più, non solo in assoluto, ma anche in proporzione.

Un individuo che produce reddito fino a 6 milioni, ad esempio, pagherà il 12% della cifra guadagnata nell'arco dell'anno.

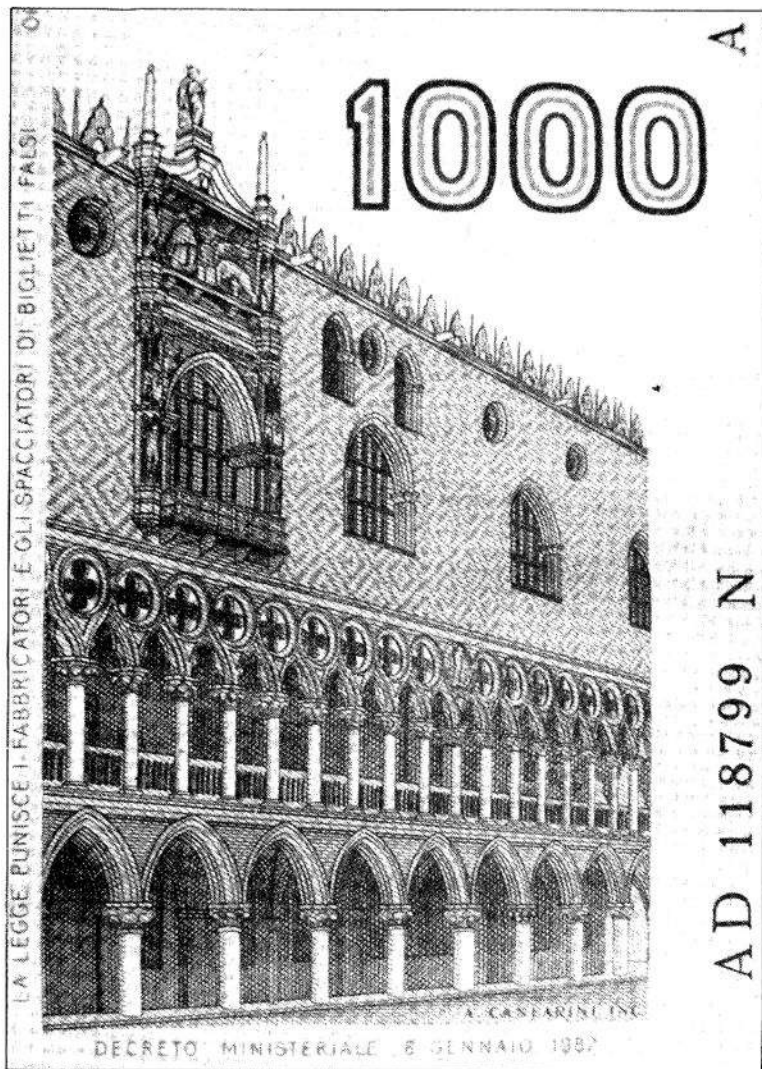
Chi ha guadagnato una cifra compresa tra 6 milioni e 11 milioni pagherà il 12% sui primi 6 milioni (cioè 720 mila lire) e il 22% sulla parte di reddito compresa tra i 6 milioni (ed una lira) fino a 11 milioni.

Chi invece (passando allo "scaglione" noto come quello del 41%) guadagna 60 milioni in un anno, pagherà 13 milioni 890 mila lire (fisse) oltre al 41%, appunto, della cifra superiore ai 50 milioni percepiti.

La tabella completa dei 9 scaglioni si può rintracciare sui fogli di istruzioni del 740, oppure esaminando il listato di queste pagine.

Il motivo della percentuale progressiva (più guadagni, più ti tasso) rientra nell'ambito di un ampio discorso di giustizia sociale e dovrebbe scoraggiare coloro che decidono di guadagnare grosse somme. Come vedremo tra breve, infatti, non sempre vale la pena darsi da fare per guadagnare cifre enormi dal momento che, alla fine, una grossa fetta andrà "persa" in tasse; scoraggiando lauti guadagni si offre la possibilità, insomma, di ripartire le varie attività lavorative tra un maggior numero di lavoratori (almeno negli intenti...).

Il programma

Il semplicissimo programma di queste pagine non solo rende automatica l'intera procedura di calcolo degli scaglioni ma, soprattutto, si presta per dissertare sul sistema di tassazione e su ciò che, alla fine, rimane nelle tasche del contribuente.

Impartendo il solito Run, lo schermo viene cancellato e compare la domanda...

Quanto hai guadagnato?

...alla quale risponderemo digitando la cifra (lorda) del nostro ipotetico contribuente.

Supponendo di rispondere: 28000000 (28 milioni), comparirà la seconda domanda...

Quanti giorni hai lavorato?

...che ci permetterà di effettuare, in seguito, alcuni ragionamenti. Per ora rispondete con il numero 231, che si riferisce al caso di un mese di vacanza (30 giorni, compresi Natale, Pasqua, ponti, viadotti, eccetera), 52 domeniche e 52 sabati (o giorni liberi in senso generale).

Appena premete il tasto Return compaiono i seguenti messaggi:

In effetti disponi di L. 21.590.000

Il modello 740 con il computer

Le note di queste pagine, pur se relative ad una "sezione" fondamentale del famoso modello 740, non esauriscono l'argomento.

Le "righe" da riempire nel modello sono talmente numerose che da qualche anno, ormai, la Systems Editoriale propone un programma, distribuito in tutte le edicole, che aiuta nella compilazione.

Anche quest'anno, quindi, potrete procurarvi, con una cifra tradizionalmente irrisoria, un fantastico programma che, aggiornato secondo le recenti disposizioni ministeriali, aiuterà tutti voi nella triste incombenza.

Occhio alle edicole, dunque, e mi raccomando... pagate le tasse!

Perchè paghi in tasse
L. 6.410.000
Ogni giorno di lavoro vale L. 93.463
Ogni giorno disponi di L. 59.150
Un oggetto che costa L. 100 lo paghi, in realtà, L. 129.689671

Subito dopo il programma è pronto per effettuare una nuova elaborazione.

Ogni giorno

Le semplici divisioni delle righe 260 e 270 consentono di scoprire che il nostro ipotetico contribuente (con reddito, ripetiamo, di 28 milioni lordi annui) vede il proprio lavoro compensato con la cifra di oltre 93 mila lire al giorno (realmente lavorativo); in media, però, dispone quotidianamente di meno di 60 mila lire.

Queste due informazioni possono essere utili per valutare, in caso di lavori (e redditi) diversi, l'effettiva convenienza di un eventuale cambio di attività. In caso di promozioni, ad esempio, si percepisce di più ma, se il posto è di una certa responsabilità, è probabile che il numero di giorni lavorativi non sia di soli 231 all'anno e non tutti di sole otto ore.

La notizia che più sconcerta, invece, è quella fornita dall'ultima elaborazione.

In pratica vi invitiamo a fare il seguente ragionamento:

Supponiamo che l'intero reddito percepito (o meglio: ciò che è rimasto detratte le tasse, vale a dire 21 milioni 590 mila lire) venga utilizzato per acquistare un'automobile che, guarda caso, costa proprio 21 milioni 590 mila lire.

In effetti l'automobile viene a costare 28 milioni perchè è stato necessario lavorare (e guadagnare) tale cifra per acquistare quell'auto: non ci interessa, insomma, che fine abbia fatto il nostro reddito; ciò che conta è che, dopo un anno ci ritroviamo senza una lira oppure, se preferite, che avete lavorato per 28 milioni per impossessarvi di un'auto che ne vale di meno.

Se, dunque, occorrono 28 milioni (lordi) allo scopo di acquistare un oggetto che ne vale di meno, quanto ci viene a costare, in realtà, un ipotetico oggeto che in vetrina vediamo prezzato con 100 lire?

La proporzione, a meno di zeri superflui, è semplice...

2159 : 2800 = 100 : X

...che "tradotta" significa: se un oggetto prezzato 2159 in realtà necessita di 2800 nostre lire-lavoro, quanto ci costerà (in termini di reddito prodotto) un oggetto che costa 100?

La risposta (oltre le 129 lire) indica che lo paghiamo più del 29%

Se, però, esaminiamo la tabella del 740, ci accorgiamo con stupore che l'aliquota dello scaglione corrispondente è quella del 27%, percentuale da applicare, tra l'altro, solo ad una parte dei 28 milioni (ed esattamente alla quota superiore agli 11 milioni).

Il "trucchetto" contabile risiede nella non reversibilità delle operazioni contenenti le percentuali; spieghiamoci meglio.

Sappiamo che se...

3 X 2 = 6

...è vero anche il viceversa:

3 = 6 : 2

Cioè è possibile "andare" da sinistra a destra (del segno eguale) e di nuovo da destra a sinistra, ottenendo il risultato originale.

Consideriamo, invece, il caso in cui si voglia determinare la somma totale ottenuta applicando il 10 % di 1000 lire:

1000 + 100 = 1100

Se, ora, "togliamo" lo stesso 10% da 1100, non otteniamo la cifra di prima (cioè 1000 lire), ma una cifra inferiore:

1100 - 110 = 990

In altre parole, andando da sinistra a destra si ottiene un risultato diverso dal procedimento contrario. Tale comportamento è proprio quello seguito nella determinazione delle tasse. Lo stato ci tassa andando da destra a sinistra (e la percentuale appare modesta) mentre invece, quando noi acquistiamo qualcosa, andiamo da sinistra a destra!

Per consolarci

Confrontando la realtà della vita e la compilazione del 740 vi può sembrare, pertanto, di essere al centro di una contabilità quantomeno disinvolta.

Ma consolatevi: se, al ristorante, il conto che vi presentano (100 mila lire) vi costringe, in realtà, a produrre un reddito di oltre 129 mila lire, pensate che se al tavolo vicino al vostro c'è Gianni Agnelli, e gli viene presentato lo stesso conto, questo gli viene a costare, in realtà, più del doppio.

Pensate, quindi, che l'Avvocato paga sempre più di voi, qualunque cosa decida di acquistare!

Come sarebbe bello pagare il triplo del conto, come fa Gianni Agnelli...

```
100 REM DETERMINAZIONE DEL VALORE REALE DEL REDDITO
110 :
120 PRINT CHR$(147);
130 PRINT: PRINTCHR$(18);:INPUT"QUANTO HAI GUADAGNATO";NL
140 INPUT"QUANTI GIORNI HAI LAVORATO";NG
150 PRINT:IF NL<6000000 THEN Y=NL*.12:GOTO 240
160 IF NL< 11000001 THEN Y=720000+   (NL-6000000)*.22:GOTO240
170 IF NL< 28000001 THEN Y=1820000+  (NL-11000000)*.27:GOTO240
180 IF NL< 50000001 THEN Y=6410000+  (NL-28000000)*.34:GOTO240
190 IF NL<100000001 THEN Y=13890000+ (NL-50000000)*.41:GOTO240
200 IF NL<150000001 THEN Y=34390000+ (NL-100000000)*.48:GOTO240
210 IF NL<300000001 THEN Y=58390000+ (NL-150000000)*.53:GOTO240
220 IF NL<600000001 THEN Y=137890000+(NL-300000000)*.58:GOTO240
230 Y=311890000+(NL-600000000)*.62
240 W=NL-Y:PRINT"IN EFFETTI DISPONI DI L."TAB(27)W
250 PRINT"PERCHE' PAGHI IN TASSE L."TAB(27)Y
260 PRINT"OGNI GIORNO DI LAVORO VALE"TAB(27)INT(W/NG)
270 PRINT"OGNI GIORNO DISPONI DI"TAB(27)INT(W/365)
280 PRINT"UN OGGETTO CHE COSTA L.100"
290 PRINT"LO PAGHI', IN EFFETTI; L."TAB(27)100*NL/W:GOTO130
300 END
```

L'EPOPEA DEL COMMODORE 64

Come è cambiato, attraverso i secoli, l'interno dell'intramontabile C/64

Dal preistorico 1982 molti chip sono passati sotto i ponti dell'informatica: nuovi computer sono nati (vissuti e morti), altri stanno per consolidare la propia immagine, altri godono ancora di ottima salute.

E' il caso, quest'ultimo, del C/64 che, con l'andar del tempo, si è adeguato ai progessi della tecnologia.

I primi tempi, tanto per riferirsi alle innovazioni più conosciute, i chip di memoria più avanzati tecnologicamente erano quelli da 8 Kram che erano considerati così "moderni" che il C/64 fu il primo computer a esser dotato di ben 64 KRam standard. Si pensi che tra breve verranno prodotti, su larghissima scala, chip Ram della capacità di un milione di bit!

Le conseguenze sul piano pratico, che più da vicino riguardano l'utente Commodore, si sono riversate positivamente sul prezzo finale.

Si è passati, infatti, dalle 990.000 lire del 1982 alle 375.000 lire odierne (listino ufficiale, I.V.A. esclusa). E' facile immaginare come, inflazione considerando, il ribasso del prezzo sia stato molto maggiore di quanto possa apparire a prima vista.

Uno dei motivi principali della diminuzione del prezzo, indubbiamente, è da attribuirsi al notevole successo di vendita del computer, che ha consentito di aumentare la produzione innescando la tradizionale corsa al ribasso.

Un motivo non meno importante è rappresentato, tuttavia, anche dalle profonde modifiche del circuito stampato originale che, nei primi modelli, occupava quasi per intero la custodia di plastica, mentre oggi si "accontenta" di riempirla per metà.

La miniaturizzazione dei circuiti integrati, e lo sviluppo delle loro potenzialità, ha contribuito alla realizzazione di un computer con una quantità irrisoria di C.I. che si è tradotta positivamente nell'abbreviazione del ciclo di fabbricazione automatica, con conseguente abbassamento dei costi di produzione.

Qui di seguito è riportato un elenco, approssimativo, sia dei componenti lasciati identici a quelli montati sui primi modelli di C/64, che dei principali progressi compiuti dalla Commodore nel corso degli anni.

Il CIA di I/O (3)

Il famoso 6526, della famiglia 65XX, è rimasto praticamente immutato, salvo esser spostato in tutt'altro punto nei moderni modelli.

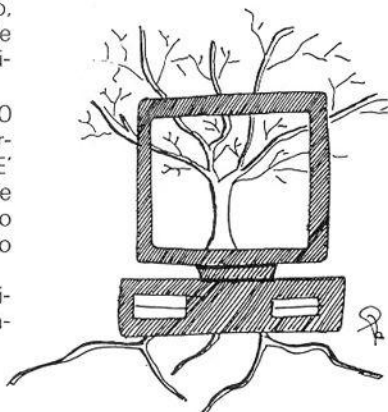
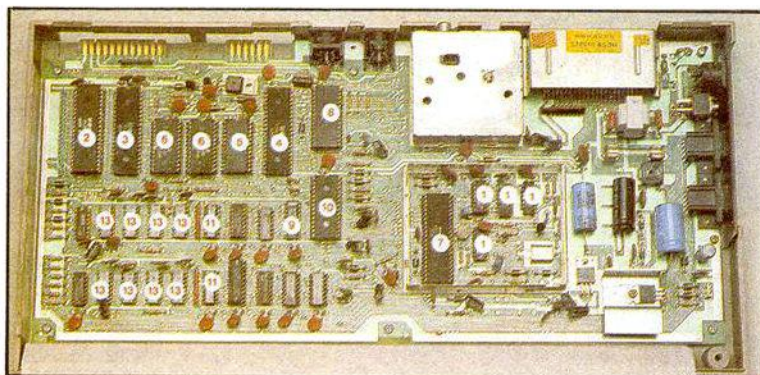
Il microprocessore (4)

Il microprocessore è sempre rimasto lo stesso (a parte il cambiamento di sigla) ma è stato leggermente spostato, per esigenze di disegno.

Rom dei caratteri (5)

E' uno dei pochi chip rimasti praticamente immutati.

Rom del Basic e del S.O. (6)

I due chip del 1982 sono stati racchiusi in un solo chip di maggiore capacità.

Video Interface Chip (7)
Appartiene alla zona del circuito stampato che più di ogni altra ha subito maggiori cambiamenti, sia come spazio occupato che come alimentazione. Il rettangolo visibile verso destra (prima versione) è stato sostituito dall'unico chip, indicato con il n.7, nell'ultima. Tale miniaturizzazione è stata possibile coustomizzando più circuiti integrati in uno solo.

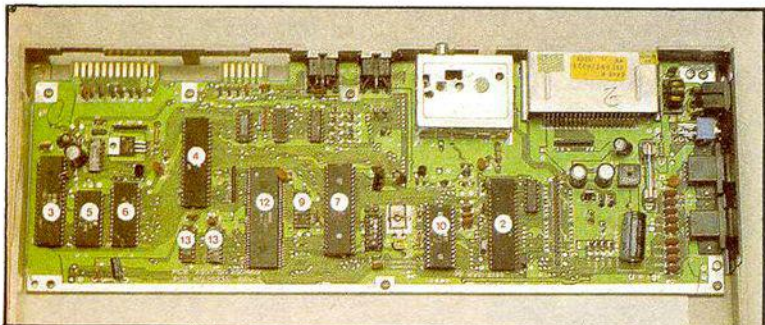
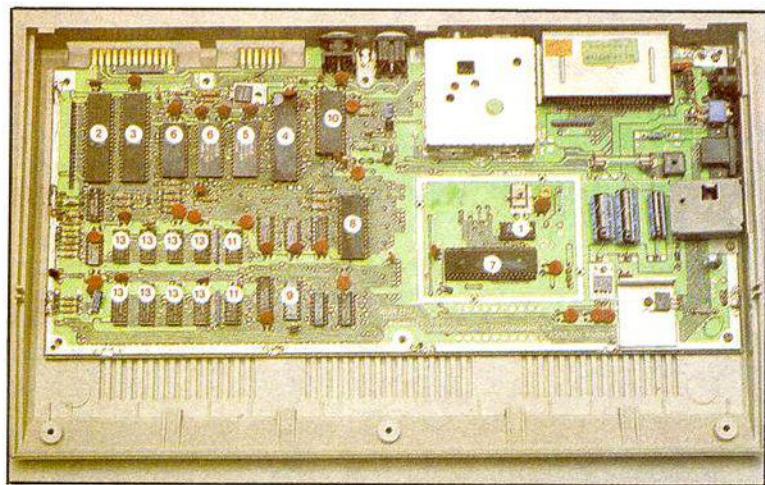
Gestione indirizzi (8)
Questo chip, di grosse dimensioni nei primi modelli, è praticamente scomparso nelle ultime schede, nel senso che le sue funzioni sono state incorporate in un chip di maggiori potenzialità.

Ram colore (9)
A causa della piccola quantità di memoria necessaria, non è stato ritenuto necessario apportare modifiche o miniaturizzazioni ulteriori.

Sound Interface Device (10)
Il famoso SID, generatore sonoro, sembra aver subito una metamorfosi solo nella sigla: da 6581 a 8580; la compatibilità tra i due, infatti, è totale.

Multiplexer (11)
Questi chip, che insieme ad altri contribuivano alla corretta gestione del calcolatore, è scomparso nelle versioni più recenti del C/64.

Multiplexer (12)
Questo chip è presente solo nelle piastre moderne e rappresenta la sintesi di altri circuiti integrati utilizzati in precedenti versioni.

Le Ram (13)
Mentre erano necessari ben 8 chip nel 1982 (in basso a sinistra nella foto), negli ultimi modelli ne troviamo soltanto due da 32 K Ram ciascuno.

(Foto tratte da "Run" ed. tedesca Okt '87)

Commodore

NEWS

COMMODORE IN TV

La recente serie di telefilm, di produzione italiana, trasmessa dal network di Berlusconi (circuito Italia 1), ha come protagonisti alcuni giovani appassionati di computer. "Cantando e ballando con Licia" è, infatti, la trasposizione in carne ed ossa della fortunata serie di cartoons dal titolo originale "Love me Licia".

Tra i protagonisti non poteva certo mancare il C/64 i cui programmi, in gran parte, sono stati scritti, adattati e "piegati" al volere del regista, dal nostro Michele Maggi, che ha trascorso un numero incredibile di ore presso gli studi di Italia 1, nel corso delle numerose registrazioni.

La trasmissione è a cadenza infrasettimanale (lunedì, mercoledì, venerdì), con inizio alle ore 20.

Un robot straordinario

Grande scalpore ha destato, nel corso della recente manifestazione fieristica specializzata in informatica (Romaufficio) un robot che parlava cinque lingue e rispondeva alle domande dei visitatori.

Dove poteva trovarsi una simile meraviglia? Ma allo stand della Commodore, diamine!

Il solito informato va dicendo in giro che, in realtà, nel robot è nascosto un ricetrasmettitore, comandato a distanza da un abile tecnico. Per saperne di più sarà necessario indagare al prossimo SMAU, ove sembra che il robot apparirà nuovamente..

Commodore Italia, ecco il nuovo manager

L'ingegner Sergio Simonelli è il nuovo Direttore Generale della Commodore Italiana. La sua provenienza da un settore professionale dell'informatica non è casuale ed indica chiaramente la decisione della Commodore International di entrare nell'attraente mercato dell'informatica di alto livello.

La Commodore ha infatti sperimentato con successo la propria presenza nel difficile ea esigente mercato tedesco, riscuotendo un risultato di tutto rispetto: ha venduto un numero di macchine (Ms-Dos) maggiore della stessa IBM e vuole tentare di ottenere analoghi risultati anche in Italia, sia nel campo Ms-Dos che Amiga.

Al nuovo direttore i nostri auguri di buon lavoro.

Un 64ista a 200 all'ora

Il giovane (appena 21enne) ed audace corridore Massimo Bonfante, sponsorizzato da Commodore, si è piazzato di recente ai primi posti in classifica, a dispetto di alcuni, sfortunati inconvenienti tecnici occorsi alla sua moto (una splendida Honda 125, appositamente elaborata). Massimo, tra l'altro, corre per conto di una marca (la Commodore, appunto) che conosce molto bene: come tantissimi altri giovani egli è, infatti, un appassionato 64ista e divide il suo tempo libero tra il computer e la moto, gli esaltanti hobby delle nuove generazioni.

I prossimi appuntamenti sportivi sono a Monza (29 maggio), Mugello (5 giugno), Misano Adriatico (10 luglio e 18 agosto).

La nostra speranza, ovviamente, è che i colori rosso e blu del marchio Commodore arrivino primi in tutte le manifestazioni!

Al momento attuale (aprile), infatti, Massimo Bonfante risulta attestato al terzo posto (su 14) con 15 punti nella classifica generale per la classe 125 monocilindriche.

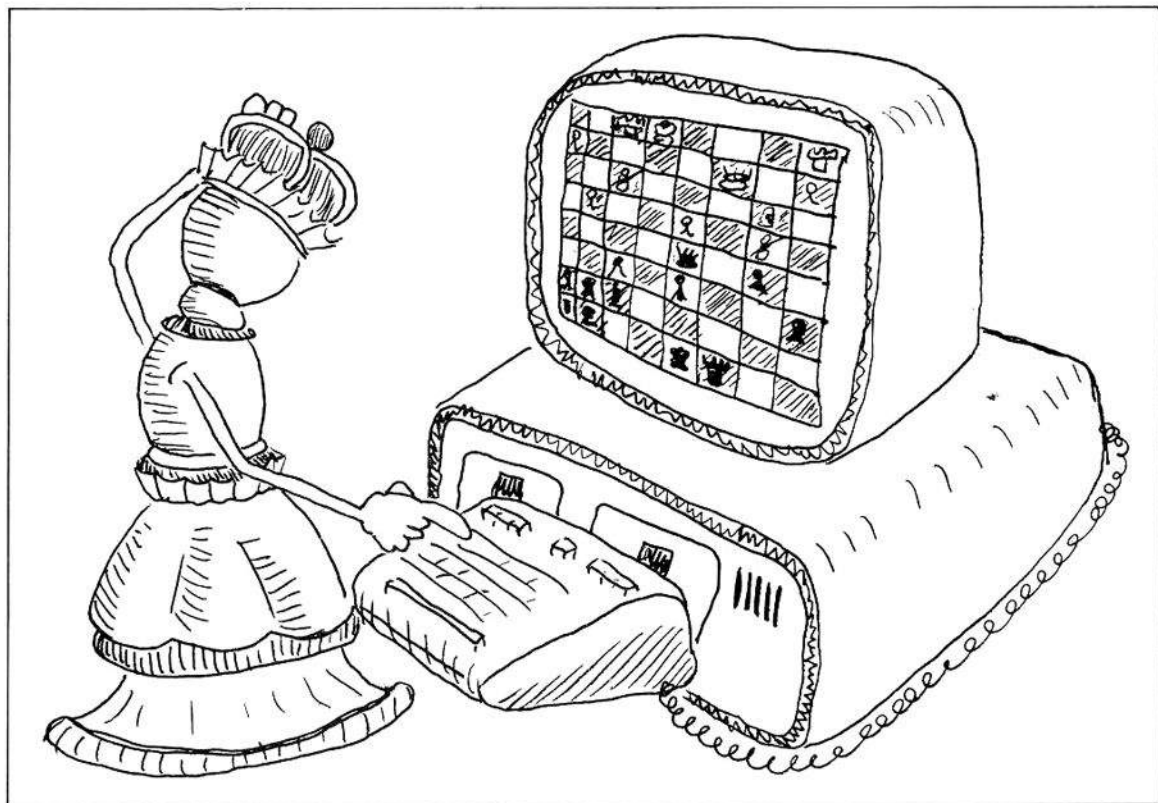
Cambio Amiga con C64

Questa straordinaria offerta non è dovuta ad un errore di stampa, ma è il risultato dell'iniziativa di cui avevamo parlato nel numero di marzo.

Presso i Commodore Point (e pochi altri punti di vendita autorizzati) sarà possibile ottenere uno sconto di ben 200.000 (duecentomila) lire, sul prezzo di listino dell'Amiga 500 in confezione speciale, consegnando al rivenditore il vecchio C/64, purchè funzionante ed utilizzabile.

Lo sconto scende a L.100.000 nel caso in cui il computer offerto in cambio sia un C/16, un Vic/20 o un Plus/4 e nel caso in cui il C/64 presentasse qualche problema (leggi: anche rotto a martellate). Consegnando un computer di marca non Commodore (purchè funzionante) lo sconto è sempre valido, pur se di L.100.000 "appena".

Perfino con un Vic/20 (oppure C/16 o Plus/4) non funzionante è possibile avere uno sconto (di L.50.000) che, pur se non esaltante, non fa certo male alla salute!

L'offerta è valida per un periodo limitato.

GUARDA COME TI SPOSTO IL VIDEO

Un programma l.m. rilocabile per effettuare lo scroll di porzioni di schermo hires indipendentemente dal linguaggio usato

di **Valentino Spataro**

Se la Computer Graphic fa passi da gigante nei sistemi più sofisticati, il buon C/64 non resta indietro.

Questo, come tutti sanno, non dispone di istruzioni Basic grazie alle quali gestire la grafica. Per questa ragione sono comparse sul mercato numerose "espansioni", in grado di supplire alla inconcepibile mancanza.

Allo sviluppo di tali espansioni non è corrisposto, però, analogo sviluppo nel campo degli effetti speciali. Le routines proposte in queste pagine si prefiggono di eliminare la carenza lamentata.

Le routine lm sono rilocabili e possono essere usate indipendentemente dalla espansione Basic presente in memoria; bisogna solo fare attenzione a non sovrapporvi le routine.

IL PROGRAMMA

Le routine l.m. sono cinque: la prima serve a considerare i parametri e disabilitare la ROM nella parte alta della memoria; le altre quattro, lanciate direttamen-

Per chi inizia

Ricordiamo, ai nuovi lettori che ci leggono per la prima volta, che la nostra rivista propone su ogni numero un gruppo di routine in Linguaggio Macchina, per il Commodore 64, interamente rilocabili. Ciò significa che l'utente potrà realizzare una vera e propria enciclopedia in L.M. personalizzata in base alle proprie esigenze.

E' ovvio che le routine proposte, per funzionare adeguatamente, soggiacciono ad alcuni limiti, ben illustrati nei primi numeri di "Nuovo Sistema".

In questa sede ci limitiamo a ricordare che, per evitare malfunzionamenti, è necessario:

- **Fissare il Top di memoria Ram a 20000.**
- **Caricare, servendosi del programma "Caricatore" a suo tempo pubblicato (o analoghi), le routine Basic contenenti le istruzioni Data.**
- **Indicare la prima locazione di memoria in cui si desidera allocare la routine stessa.**
- **Attivare la routine secondo i suggerimenti indicati nei Demo a corredo.**

te dalla prima, servono per lo scroll del video nelle quattro direzioni. La sintassi dell'istruzione è...

SYS LOC, X, Y, LX, LY, MODE

...dove...

- LOC è la locazione di partenza della prima routine (quella di input).
- X, Y sono le coordinate dell'angolo superiore sinistro della porzione di schermo desiderata (considerato di grandezza 25x40).
- LX, LY sono le rispettive dimensioni.
- MODE indica la locazione di partenza della routine di spostamento che si desidera richiamare.

Per non avere il problema di ricordare gli start address (locazioni di partenza) delle routine l.m, il listato Basic di caricamento provvede ad assegnare alle variabili ALt, BAsso, DEstra, SNistra i rispettivi start address. Da notare che SNISTRA e ALT non sono errori di stampa, ma necessari rimedi ai syntax error in cui si incorrerebbe usando ALTO e SINISTRA (parole chiave: TO, SIN).

Per spostare quindi a sinistra di un pixel una porzione di schermo basterà:

SYS LOC, X, Y, LX, LY, SINISTRA

Osservando le linee DATA potrete notare che le prime due informazioni di ogni routine rappresentano lo start address e la lunghezza della medesima. Variando a piacere lo start address (e tenendo presente la lunghezza della routine per evitare sovrapposizioni) potrete rilocarle; gli start address contenuti nelle variabili saranno automaticamente aggiornati. Inoltre la variabile LOC conterrà lo start address della routine di input.

IL PROGRAMMA LM

Ecco qui di seguito alcune informazioni per completare le spiegazioni del disassemblato.

Le routine provvedono a spostare di un pixel i disegni (o scritte, con piacevoli effetti di scorrimento) contenuti nella porzione di schermo hi-res. Nel caso di spostamenti verticali (cioè verso l'alto e il basso) lo spostamento di un pixel consiste nello spostamento di un byte da una posizione inferiore a quella superiore o viceversa.

Diversamente, per i movimenti orizzontali, si deve ricorrere allo spostamento dei bit all'interno di ogni byte facendo passare i bit in sovrappiù nel byte a lato.

Facciamo un esempio: dati i due byte affiancati...

0000-1000 0110-0000

...ed operando uno spostamento a destra si ottiene:

C000-0100 0011-0000

dove C sarà 0 oppure 1 a seconda dello status del bit immediatamente a sinistra.

Come potete intuire, tutto questo è più o meno difficile a seconda della disposizione dei byte nella bitmap. Data per saputa questa informazione, basterà aggiungere che le locazioni 4, 5, 6, 7 si incaricano di puntare ai byte da cui prendere le informazioni e ai byte dove mettere le informazioni opportunamente trattate.

Osservando il disassemblato i più esperti avranno sicuramente trovato l'istruzione JMP e si saranno chiesti come conciliarsi con la rilocabilità.

In effetti tutte le routine di scroll terminano con un JMP (START): START è un puntatore alla routine di abilitazione del kernal contenuta nello stessa routine di input e in particolare 3 byte dopo lo start address di questa routine. Ora, sapendo che quando si dà una SYS a una locazione, questa viene memorizzata nelle locazioni $14 e $15, basterà salvare tali valori addizionati a 3 in START e START+1 per far saltare poi il programma a questa procedura.

La procedura che ha richiamato tale modo di opera-

re è quella di riabilitazione della ROM del Kernal: la pagina grafica utilizzata infatti è quella residente sotto la ROM del Kernal: per prelevare i dati della RAM sottostante bisogna disattivare la ROM, come eseguito da programma; evidentemente se alla fine della procedura non si riattiva la ROM e si torna al Basic il computer non sarà più in grado di svolgere alcuna istruzione Basic. Essendo questa una procedura indispensabile per le quattro routine di spostamento è stato usato l'artificio sopra descritto per risparmiare byte.

L'INTERPRETE USATO

Il programma di queste pagine gira se, in precedenza, è stato caricato ed attivato il simulatore Gw-Basic.

Le routine l.m. tuttavia, funzionano perfettamente anche con altri interpreti (o in loro assenza) purchè, ovviamente, la pagina grafica sia la stessa di quella usata dal Gw-Basic.

UN NUOVO CONTROLLO

Il controllo della corretta digitazione, come è noto, è uno dei problemi più scottanti per gli appassionati.

Di solito, nei programmi l.m. che pubblichiamo, è sempre presente un controllo che, però, si limita a tener conto della somma dei vari Data.

Se, pertanto, invece di scrivere...

1000 Data 169, 35, XXXX

...si digita...

1000 Data 168, 36, XXXX

...il risultato globale (checksum) è sempre lo stesso ed il semplicistico controllo non è efficace.

Da questo numero di C.C.C. pubblicheremo, soprattutto per le routine più lunghe, un ulteriore controllo che non solo effettua il conteggio, come al solito, ma determina le semisomme dei byte pari e dispari.

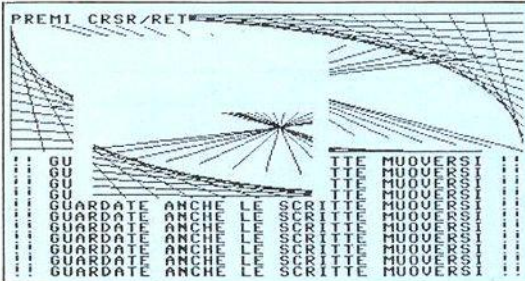
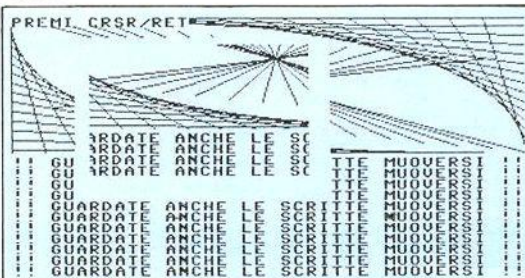
Per usare il triplo controllo è necessario dapprima attivare la routine che, tramite Read e Data, trascrive i codici macchina in memoria. Poi, prima di impartire la SYS (altrimenti, in caso di errore, buonanotte!) caricare la brevissima routine Basic che, dimenticavamo di dirlo, controlla il reale contenuto della memoria e non la lettura dei Data.

Se non compare nessun messaggio di errore, la probabilità di corretta trascrizione è elevatissima. A seconda, invece, dell'errore segnalato, dovrete controllare i byte pari, dispari, o... tutti!

Si noti che la routine è numerata da 63000 in poi. Nulla di male, quindi, nell'eseguire un Append al programma caricatore, in modo da avere un unico controllo.

```
200 REM ********************************
210 REM * SCROLL IN HIRES. ROUTINES    *
220 REM * RILOCABILI E INDIPENDENTI DA*
230 REM * ESPANSIONE BASIC UTILIZZATA *
240 REM * BY SPATARO VALENTINO.        *
250 REM * IL DEMO PROPOSTO RICHIEDE    *
260 REM * GW-BASIC: SE V2.0 NON SONO   *
261 REM * NECESSARIE MODIFICHE         *
262 REM * SE V1.0 CAMBIARE LOCATE Y,X *
263 REM * IN LOCATE (X*8,Y*8)          *
270 REM ********************************
280 POKE55,0:POKE56,97:CLR:REM FISSA TOP MEMO. NECESSARIO SOLO PER IL DEMO
290 GOSUB1000:REM CARICA LE ROUTINES
300 X=5:Y=2:LX=20:LY=15
310 SCREEN1:COLOR0,1,0:CLS1:GOSUB450
320 LOCATE1,1 :PRINT"PREMI CRSR/RET"
330 GETA$:Q=ASC(A$+CHR$(0))
340 IFQ=29 THENSYSLOC,X,Y,LX,LY,DESTRA
350 IFQ=157THENSYSLOC,X,Y,LX,LY,SNISTRA
360 IFQ=145THENSYSLOC,X,Y,LX,LY,ALT
370 IFQ=17 THENSYSLOC,X,Y,LX,LY,BASSO
380 IFQ=13THENGOSUB400
390 GOTO330
400 SCREEN0:PRINTCHR$(147)
410 INPUT"X,Y,LX,LY";X,Y,LX,LY
420 IFX>40ORY>25THENPRINT"ERRORE NELLE COORDINATE X O Y":GOTO410
430 IFLX+X>40ORY+LY>25THENPRINT"ERRORE NELLE COORDINATE LX O LY":GOTO410
440 GOTO310
450 A$="!! GUARDATE ANCHE LE SCRITTE MUOVERSI !!"
460 FORA=14TO24:LOCATE A,1:PRINTA$:NEXT
470 FORA=0TO100STEP7:LINE (A*3,0)-(320,A)
480 LINE (A*3,100)-(0,A)
```

```
490 LINE (A*3,A/3)-(320-A*3,100-A/3):NEXT:RETURN
1000 Q-0:PRINTCHR$(147)"N.BYTES     N.ROUTINE   START A.  END A."
1010 Q-Q+1:READA1,A2:GOSUB1060:CK-0:PRINTTAB(20)Q;A1;A1+A2CHR$(145)
1020 FORA-0TOA2:READA3:POKEA1+A,A3:CK-CK+A3
1030 PRINTA2"/"A" "CHR$(145):NEXT:READA3:IFCK<>A3THENPRINT"ERROR IN GRUPPO "Q:EN
D
1040 PRINT:IFQ<>5GOTO1010
1050 PRINT"ATTIVATE":RETURN
1060 IFQ-1THENLOC-A1
1070 IFQ-2THENSNISTRA-A1
1080 IFQ-3THENBASSO-A1
1090 IFQ-4THENALT-A1
1100 IFQ-5THENDESTRA-A1
1110 RETURN
1120 PRINT"SOVRAPPOSIZIONE DELLE ROUTINES: RIDEFI- NIRE GLI START ADDRESS":END
20000 DATA 25000,164:REM ROUTINE N.1 DI INPUT
20001 DATA 024,144,016,169,003,005,001
20002 DATA 133,001,169,001,013,014,220
20004 DATA 141,014,220,088,096,165,020
20006 DATA 024,105,003,133,251,144,002
20008 DATA 230,021,165,021,133,252,032
20010 DATA 241,183,224,039,016,119,142
20012 DATA 194,002,032,241,183,224,024
20014 DATA 016,109,142,195,002,032,241
20016 DATA 183,138,109,194,002,201,042
20018 DATA 016,095,134,002,032,241,183
20020 DATA 138,109,195,002,201,027,016
20022 DATA 082,134,003,162,000,134,253
20024 DATA 169,024,133,254,172,195,002
20026 DATA 192,000,240,017,230,254,169
20028 DATA 064,024,101,253,133,253,144
20030 DATA 002,230,254,136,024,144,235
20032 DATA 172,194,002,192,000,240,015
20034 DATA 169,008,024,101,253,133,253
20036 DATA 144,002,230,254,136,024,144
20038 DATA 237,032,253,174,032,138,173
20040 DATA 032,247,183,169,254,045,014
20042 DATA 220,141,014,220,169,252,037
20044 DATA 001,133,001,108,020,000,162
20046 DATA 014,108,000,003,19326
20047 DATA 25170,105:REM ROUTINE N.2:SINISTRA.
20048 DATA 024,165,253,133,004,165,254,133,005,133
20050 DATA 007,169,008,101,004,133,006
20052 DATA 144,002,230,007,166,002,160
20054 DATA 000,224,001,208,003,024,144
20056 DATA 004,024,177,006,010,177,004
20058 DATA 042,145,004,200,192,008,208
20060 DATA 235,165,006,133,004,165,007
20062 DATA 133,005,152,024,101,006,133
20064 DATA 006,144,002,230,007,224,000
20066 DATA 202,234,234,208,208,160,000
20068 DATA 177,006,010,200,192,008,208
20070 DATA 248,165,003,198,003,240,016
20072 DATA 024,230,254,169,064,101,253
20074 DATA 133,253,144,002,230,254,024
20076 DATA 144,153,108,251,000,11806
20077 DATA 25280,130:REM RUUTINE 3: GIU'
20078 DATA 166,003,202,224,000,240,016,230,254
20080 DATA 024,169,064,101,253,133,253
20082 DATA 144,002,230,254,024,144,235
20084 DATA 156,002,165,253,133,004,165
20086 DATA 254,133,005,133,007,198,007
20088 DATA 024,165,004,233,063,133,006
20090 DATA 176,002,198,007,160,007,136
20092 DATA 177,004,200,145,004,136,192
20094 DATA 000,208,245,160,007,165,003
20096 DATA 201,001,208,004,169,000,240
20098 DATA 002,177,006,160,000,145,004
20100 DATA 024,169,008,101,004,133,004
20102 DATA 144,002,230,005,202,224,000
20104 DATA 240,011,165,004,133,006,165
```

```
20106 DATA 005,133,007,024,144,185,198
20108 DATA 003,165,003,240,016,198,254
20110 DATA 024,165,253,233,063,133,253
20112 DATA 176,002,198,254,024,144,151
20114 DATA 108,251,000,15248
20115 DATA 25420,104:REM ROUTINE N.4:ALTO
20116 DATA 166,002,165,253,133,004,165,254,133,005,133
20118 DATA 007,230,007,024,169,064,101
20120 DATA 004,133,006,144,002,230,007
20122 DATA 160,000,200,177,004,136,145
20124 DATA 004,200,192,007,208,245,160
20126 DATA 000,165,003,201,001,208,004
20128 DATA 169,000,240,002,177,006,160
20130 DATA 007,145,004,024,169,008,101
20132 DATA 004,133,004,144,002,230,005
20134 DATA 165,004,133,006,165,005,133
20136 DATA 007,202,224,000,208,188,198
20138 DATA 003,165,003,240,016,024,230
20140 DATA 254,169,064,101,253,133,253
20142 DATA 144,002,230,254,024,144,154
20144 DATA 108,251,000,11552
20145 DATA 25540,95:REM ROUTINE N.5: DESTRA
20146 DATA 164,002,152,170
20147 DATA 136,192,000,240,014,169,008
20148 DATA 024,101,253,133,253,144,002
20150 DATA 230,254,024,144,237,165,254
20152 DATA 133,005,165,253,024,233,007
20154 DATA 133,004,176,002,198,005,160
20156 DATA 000,224,001,024,240,003,177
20158 DATA 004,074,177,253,106,145,253
20160 DATA 200,192,008,208,238,165,004
20162 DATA 133,253,165,005,133,254,202
20164 DATA 224,000,208,212,198,003,165
20166 DATA 003,240,016,230,254,169,072
20168 DATA 024,101,253,133,253,144,002
20170 DATA 230,254,024,144,163,108,251,000,12882
```

```
63000 REM TRIPLO CONTROLLO ERRORI
63010 DA- 25000: A-25164: REM LOCAZIONI DI INIZIO E DI FINE
63020 Y1- 8762: Y2-10564: REM SEMISOMMA PARI E DISPARI
63025 Y3- 19326: REM SOMMA TOTALE
```

```
63000 REM TRIPLO CONTROLLO ERRORI
63010 DA- 25170: A-25275: REM LOCAZIONI DI INIZIO E DI FINE
63020 Y1- 5520: Y2- 6286: REM SEMISOMMA PARI E DISPARI
63025 Y3- 11806: REM SOMMA TOTALE
```

```
63000 REM TRIPLO CONTROLLO ERRORI
63010 DA- 25280: A-25410: REM LOCAZIONI DI INIZIO E DI FINE
63020 Y1- 7022: Y2- 8226: REM SEMISOMMA PARI E DISPARI
63025 Y3- 15248: REM SOMMA TOTALE
```

```
63000 REM TRIPLO CONTROLLO ERRORI
63010 DA- 25420: A-25524: REM LOCAZIONI DI INIZIO E DI FINE
63020 Y1- 6034: Y2- 5518: REM SEMISOMMA PARI E DISPARI
63025 Y3- 11552: REM SOMMA TOTALE
```

```
63000 REM TRIPLO CONTROLLO ERRORI
63010 DA- 25540: A-25635: REM LOCAZIONI DI INIZIO E DI FINE
63020 Y1- 6350: Y2- 6532: REM SEMISOMMA PARI E DISPARI
63025 Y3- 12882: REM SOMMA TOTALE
63030 FOR I-DA TO A STEP 2:X1-X1+PEEK(I):NEXT
63040 FOR I-DA+1 TO A STEP 2:X2-X2+PEEK(I):NEXT
63045 FOR I-DA TO A :X3-X3+PEEK(I):NEXT
63050 IF Y1<>X1 THEN PRINT"ERRORE DISPARI"
63060 IF Y2<>X2 THEN PRINT"ERRORE PARI"
63070 IF Y3<>X3 THEN PRINT"ERRORE TOTALE"
```

```
 1 X        -      $02C2
 2 Y        -      $02C3
 3 LX       -      $02
 4 LY       -      $03
 5 INPUT    -      $B7F1
 6 INPUT1   -      $AEFD
 7 INPUT2   -      $AD8A
 8 INPUT3   -      $B7F7
 9 BITMAP   -      $FD
10 START    -      $FB
11 IERROR   -      $0300
12 ********************************
13 * SCROLL                       *
14 * SYS INPUT,X,Y,LX,LY,TYPEADDRESS
15 ********************************
16 *    ABILITA ROM BASIC E KERNAL *
17 ********************************
18             ORG    30000
19             CLC
20             BCC    INIZIO
21 ABILITA     LDA    #$03
22             ORA    $01
23             STA    $01
24             LDA    #$01
25             ORA    $DC0E
26             STA    $DC0E
27             CLI
28             RTS
29 ********************************
30 * INPUT DATI                   *
31 ********************************
32 INIZIO      LDA    $14          ;FISSA PUNTATORE A
33             CLC                 ;SUBROUTINE DI
34             ADC    #03          ;ABILITAZIONE ROM
35             STA    START
36             BCC    SALTO
37             INC    $15
38 SALTO       LDA    $15
39             STA    START+1
40             JSR    INPUT
41             CPX    #39
```

```
42              BPL     MISTAKE
43              STX     X            ;PRENDE X
44              JSR     INPUT
45              CPX     #24
46              BPL     MISTAKE
47              STX     Y            ;PRENDE Y
48              JSR     INPUT
49              TXA
50              ADC     X
51              CMP     #41
52              BPL     MISTAKE
53              STX     LX           ;PRENDE LX
54              JSR     INPUT
55              TXA
56              ADC     Y
57              CMP     #26
58              BPL     MISTAKE
59              STX     LY           ;PRENDE LY
60 * DETERMINA LOC INIZIO BITMAP
61              LDX     #$00
62              STX     BITMAP
63              LDA     #$E0
64              STA     BITMAP+1
65              LDY     Y
66 LOOP30       CPY     #00
67              BEQ     LOOP42
68 LOOP40       INC     BITMAP+1
69              LDA     #$40
70              CLC
71              ADC     BITMAP
72              STA     BITMAP
73              BCC     LOOP41
74              INC     BITMAP+1
75 LOOP41       DEY
76              CLC
77              BCC     LOOP30
78 LOOP42       LDY     X
79 LOOP43       CPY     #00
80              BEQ     DISAB
81 LOOP44       LDA     #08
82              CLC
83              ADC     BITMAP
84              STA     BITMAP
85              BCC     LOOP45
86              INC     BITMAP+1
87 LOOP45       DEY
88              CLC
89              BCC     LOOP43
90 ********************************
91 * DISABILITA ROM BASIC E KERNAL
92 ********************************
93 DISAB        JSR     INPUT1
94              JSR     INPUT2
95              JSR     INPUT3
96              LDA     #$FE
97              AND     $DC0E
98              STA     $DC0E
99              LDA     #$FC
100             AND     $01
101             STA     $01
102             JMP     ($014)
103 MISTAKE     LDX     #$0E
104             JMP     (IERROR)
105 ********************************
106 * LEFT                         *
107 ********************************
108 LEFT        CLC
109             LDA     BITMAP       ;IMPOSTA I PUNTATORI:
110             STA     $04          ;$04 E $05 A BYTE CORRENTE
111             LDA     BITMAP+1     ;$06 E $07 AL CORRISPON-
112             STA     $05          ;DENTE BYTE A DESTRA DA
113             STA     $07          ;CUI PRENDERE 1 PIXEL
114             LDA     #$08
115             ADC     $04
116             STA     $06
117             BCC     SALTO1
118             INC     $07
119 SALTO1      LDX     LX
120 LOOP1       LDY     #$00
121 LOOP2       CPX     #01
122             BNE     LOOP3        ;NON PRENDE IL PIXEL DA
123             CLC                  ;DESTRA SE E' L'ULTIMA
124             BCC     LOOP4        ;COLONNA A DESTRA
125 LOOP3       CLC
126             LDA     ($06),Y
127             ASL
128 LOOP4       LDA     ($04),Y
129             ROL
130             STA     ($04),Y
131             INY
132             CPY     #$08
133             BNE     LOOP2
134             LDA     $06
135             STA     $04
136             LDA     $07
137             STA     $05
138             TYA
139             CLC
140             ADC     $06
141             STA     $06
142             BCC     SALTO2
143             INC     $07
144 SALTO2      CPX     #$00
145             DEX
146             NOP
147             NOP
148             BNE     LOOP1
149             LDY     #$00
150 LOOP5       LDA     ($06),Y
151             ASL
152             INY
153             CPY     #$08
154             BNE     LOOP5
155             LDA     LY
156             DEC     LY
157             BEQ     ENDLEFT
158             CLC
159             INC     BITMAP+1
160             LDA     #$40
161             ADC     BITMAP
162             STA     BITMAP
163             BCC     SALTO3
164             INC     BITMAP+1
165 SALTO3      CLC
166             BCC     LEFT
167 ENDLEFT     JMP     (START)
168 ********************************
169 * DOWN                         *
170 ********************************
171             LDX     LY
172 LOOP10      DEX
173             CPX     #00
174             BEQ     DOWN
175             INC     BITMAP+1
176             CLC
177             LDA     #$40
178             ADC     BITMAP
179             STA     BITMAP
```

```
180            BCC    LOOP11
181            INC    BITMAP+1
182 LOOP11     CLC
183            BCC    LOOP10
184 DOWN       LDX    LX
185            LDA    BITMAP
186            STA    $04
187            LDA    BITMAP+1
188            STA    $05
189            STA    $07
190 LOOP12     DEC    $07
191            CLC
192            LDA    $04
193            SBC    #63
194            STA    $06
195            BCS    LOOP13
196            DEC    $07
197 LOOP13     LDY    #$07
198 LOOP14     DEY
199            LDA    ($04),Y
200            INY
201            STA    ($04),Y
202            DEY
203            CPY    #$00
204            BNE    LOOP14
205            LDY    #$07
206            LDA    LY
207            CMP    #$01
208            BNE    LOOP15
209            LDA    #$00
210            BEQ    LOOP16
211 LOOP15     LDA    ($06),Y
212 LOOP16     LDY    #$00
213            STA    ($04),Y
214            CLC
215            LDA    #$08
216            ADC    $04
217            STA    $04
218            BCC    LOOP17
219            INC    $05
220 LOOP17     DEX
221            CPX    #00
222            BEQ    LOOP18
223            LDA    $04
224            STA    $06
225            LDA    $05
226            STA    $07
227            CLC
228            BCC    LOOP12
229 LOOP18     DEC    LY
230            LDA    LY
231            BEQ    ENDDOWN
232            DEC    BITMAP+1
233            CLC
234            LDA    BITMAP
235            SBC    #63
236            STA    BITMAP
237            BCS    LOOP19
238            DEC    BITMAP+1
239 LOOP19     CLC
240            BCC    DOWN
241 ENDDOWN    JMP    (START)
242 *********************************
243 * UP                            *
244 *********************************
245 UP         LDX    LX             ;METTE .N X IN NUMERO
246            LDA    BITMAP         ;DI COLONNE. $04 E $05
247            STA    $04            :PUNTANO AI BYTES CORRENTI
248            LDA    BITMAP+1       ;E $06 E $07 AL BYTE SOTTO-
249            STA    $05            ;STANTE AD ESSI
250            STA    $07
251 LOOP20     INC    $07
252            CLC
253            LDA    #$40
254            ADC    $04
255            STA    $06
256            BCC    LOOP21
257            INC    $07
258 LOOP21     LDY    #$00
259 LOOP22     INY
260            LDA    ($04),Y
261            DEY
262            STA    ($04),Y
263            INY
264            CPY    #$07
265            BNE    LOOP22
266            LDY    #$00
267            LDA    LY             ;PRENDE IL BYTE SOTTOSTANTE
268            CMP    #$01           ;SOLO SE NON SI TRATTA
269            BNE    LOOP23         ; DELL'ULTIMA RIGA
270            LDA    #$00
271            BEQ    SALTO30
272 LOOP23     LDA    ($06),Y
273 SALTO30    LDY    #$07
274            STA    ($04),Y
275            CLC
276            LDA    #$08           ;SPOSTA I PUNTATORI
277            ADC    $04            ;ALLA COLONNA SUCCESSIVA
278            STA    $04
279            BCC    LOOP24
280            INC    $05
281 LOOP24     LDA    $04
282            STA    $06
283            LDA    $05
284            STA    $07
285            DEX
286            CPX    #00
287            BNE    LOOP20
288            DEC    LY
289            LDA    LY
290            BEQ    ENDUP
291            CLC                   ;PUNTA ALLA RIGA INFERIORE
292            INC    BITMAP+1
293            LDA    #$40
294            ADC    BITMAP
295            STA    BITMAP
296            BCC    LOOP25
297            INC    BITMAP+1
298 LOOP25     CLC
299            BCC    UP
300 ENDUP      JMP    (START)
301 *********************************
302 * RIGHT                         *
303 *********************************
304 RIGHT      LDY    LX             ;SI POSIZIONA SULLA
305            TYA                   ;ULTIMA COLONNA
306            TAX                   ;DI OGNI RIGA
307 LOOP39     DEY                   ;E METTE IN X IL
308            CPY    #00            ;NUMERO DI COLONNE
309            BEQ    LOOP31
310            LDA    #08
311            CLC
312            ADC    BITMAP
313            STA    BITMAP
314            BCC    SALTO20
315            INC    BITMAP+1
316 SALTO20    CLC
317            BCC    LOOP39
318 LOOP31     LDA    BITMAP+1       ;IMPOSTA PUNTATORI
319            STA    $05            ;$04 E $05 SUI BYTES
320 LOOP33     LDA    BITMAP         ;A SINISTRA DI QUELLI
321            CLC                   ;PUNTATI DA BITMAP
322            SBC    #07
323            STA    $04
324            BCS    SALTO23
325            DEC    $05
326 SALTO23    LDY    #00            ;PRENDE UN BIT
327 LOOP34     CPX    #01            ;DA SINISTRA E LO
328            CLC                   ;METTE A DESTRA
329            BEQ    LOOP32         ;PERO' SE LA COLONNA
330            LDA    ($04),Y        ;CONSIDERATA E' LA
331            LSR                   ;PRIMA A SINISTRA
332 LOOP32     LDA    (BITMAP),Y     ;ALLORA C=0
333            ROR
334            STA    (BITMAP),Y
335            INY
336            CPY    #08
337            BNE    LOOP34
338            LDA    $04            ;SPOSTA I PUNTATORI
339            STA    BITMAP         ;8 BYTES PIU' INDIETRO
340            LDA    $05
341            STA    BITMAP+1
342            DEX
343            CPX    #00
344            BNE    LOOP33
345            DEC    LY             ;DECREMENTA IL NUMERO
346            LDA    LY             ;DI RIGHE E SE NON E'
347            BEQ    ENDRIGHT       ;FINITO
348            INC    BITMAP+1       ;AGGIORNA I
349            LDA    #72            ;PUNTATORI ALLA RIGA
350            CLC                   ;SUCCESSIVA
351            ADC    BITMAP
352            STA    BITMAP
353            BCC    SALTO22
354            INC    BITMAP+1
355 SALTO22    CLC
356            BCC    RIGHT
357 ENDRIGHT   JMP    (START)
```

ALLA SC
DEGLI IS
CON I
COMM

Una delle principali applicazioni
nel campo grafico, è certamente
A volte, però, si presentano a

Tutti dovrebbero sapere, almeno ce lo auguriamo, che cosa sono gli istogrammi.

L'uomo riesce a comprendere meglio la consistenza di alcuni dati se questi, anzichè esser presentati sottoforma di valori numerici, sono rappresentati ricorrendo a semplici figure geometriche, di dimesioni direttamente proporzionali ai valori stessi.

In effetti, da un punto di vista puramente teorico, non c'è alcuna differenza tra le due rappresentazioni (numeriche e geometriche); è innegabile, tuttavia, la constatazione che, con un colpo d'occhio, si ha subito l'idea delle grandezze con cui si ha a che fare.

La più semplice rappresentazione geometrica è sicuramente l'istogramma che simboleggia una grandezza numerica mediante un rettangolo di altezza ad essa proporzionale.

Posizionando vicini tra loro più istogrammi, si evidenzia subito la differenza esistente; a pensarci bene, però, sarebbero sufficienti non dei rettangoli, ma vari segmenti di lun-

OPERTA
OGRAMMI
PICCOLI"
ODORE

chieste ad un computer, operando visualizzazione degli istogrammi. uni problemi, la cui soluzione...

```
100 REM ISTOGRAMMI PROPORZIONALI
110 REM C/128, C/16, PLUS/4
120 :
130 FOR S=1 TO 20:REM CICLO DI 20 GRUPPI
140 GRAPHIC 2,1:  REM DI ISTOGRAMMI CASUALI
150 FOR X=1 TO 319 STEP 8*S:REM S=LARGHEZZA
160 Y=RND(0)*150:         REM ORDINATA CASUALE
170 BOX 1,X,Y,X+6*S,150:REM RETTANGOLI VUOTI
180 CHAR 1,Z*S,19,CHR$(65+Z):REM CARATTERE
190 Z=Z+1:NEXT X:Z=0:REM CHIUSURA CICLI
200 FOR UU=1 TO 300:NEXT UU:REM RITARDO
210 NEXT S: REM CHIUSURA ULTIMO CICLO
220 END
```

ghezza proporzionale alle grandezze in questione.

Rimane il fatto, però, che un rettangolo, grazie al suo "spessore", comunica più in fretta la grandezza che rappresenta.

Non sempre, però, la larghezza di un istogramma "deve" essere grande, e lo dimostra il primo programma, in grado di generare, su una sola schermata in alta risoluzione, ben 40 colonne, separate tra loro di al-

cuni pixel e larghe 8 ciascuna.

Al di sotto di ciascun istogramma viene anche riportato un carattere alfanumerico che può essere utile per individuare la grandezza cui si riferisce l'istogramma stesso.

Si può notare facilmente, infatti, che il programma genera ben 20 gruppi di istogrammi in altrettante schermate; ogni volta che viene generata una nuova schermata il numero di valori rappresentabili diminuisce, a tutto vantaggio della larghezza degli stessi (che aumenta a mano a mano) e della conseguente leggibilità. I vari caratteri alfanumerici sono posizionati sempre con un certo ordine.

Da notare che la base degli istogrammi non coincide con la base dello schermo dal momento che si è attivato il modo grafico n.2 che, come è noto, lascia in modo testo le ultime righe dello schermo per favorire la comparsa di messaggi che non disturbino la schermata hi-res.

Volendo sfruttare l'intera dimensione dello schermo (ci riferiamo, ovviamente, ai computer C/128, C/16 e Plus/4) sarà sufficiente cambiare in Graphic 1,1 il comando di riga 140, digitare il valore 199 al posto di 150 (riga 170) e modificare il 19 di riga 180 in 24.

Inutile ricordare che l'istogramma viene disegnato a partire dall'alto verso il basso e di questo fatto è necessario tener conto nella rappresentazione di valori numerici.

Il secondo programma ("Istogrammi sovrapposti") è una delle soluzioni di un problema classico: la sovrapposizione di istogrammi relativi a dati omogenei.

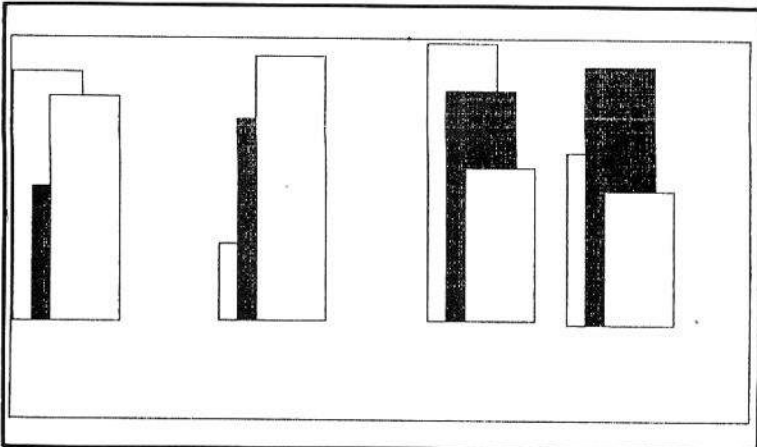
Supponiamo, ad esempio, di voler conoscere l'andamento delle vendite negli ultimi tre anni. Si potrebbero tracciare 36 istogrammi (3 x 12 mesi) relativi, appunto, ai 36 mesi considerati, ma certamemte è più utile rapportare tra loro i valori dei tre mesi di gennaio, dei tre di febbraio e così via.

Di solito questo tipo di problema si risolve ricorrendo ad istogrammi tridimensionali visualizzati in assonometria, di notevole effetto grafico e

```
100 REM ISTOGRAMMI SOVRAPPOSTI
110 REM C/128, C/16, PLUS/4
120 PRINT CHR$(147):REM CANCELLA SCHERMO
130 :
140 GRAPHIC 0,0:PRINTCHR$(19);:REM TESTO
150 INPUT"AMPIEZZA";AM:REM LARGH.ISTOGRAMMA
160 INPUT"SFALSAM.";SF:REM SFALSAMENTO ISTOG
170 INPUT"ASCISSA 1";XA(1):REM POSIZIONE
180 REM POSIZIONAM. AUTOMATICO DELLE ASCISSE
190 XB(1)=XA(1)+AM:REM RIFERIMENTO PRIMARIO
200 XA(2)=XA(1)+SF:XB(2)=XA(2)+AM
210 XA(3)=XA(2)+SF:XB(3)=XA(3)+AM
220 REM INPUT DELLE ORDINATE CORRISPONDENTI
230 INPUT"ORDINATA 1";YA(1)
240 INPUT"ORDINATA 2";YA(2)
250 INPUT"ORDINATA 3";YA(3)
260 GRAPHIC1,1:REM MODO GRAFICO HI-RES
270 REM PRIMO ISTOGRAMMA VUOTO
280 BOX 1,XA(1),YA(1),XB(1),150
290 REM SECONDO ISTOGRAMMA "PIENO"
300 BOX 1,XA(2),YA(2),XB(2),150,,1
310 REM TERZO ISTOGRAMMA "CANCELLATO"
320 BOX 0,XA(3),YA(3),XB(3),150,,1
330 REM TERZO ISTOGRAMMA VUOTO
340 BOX 1,XA(3),YA(3),XB(3),150
350 GETKEY A$:REM ATTESA PRESSIONE TASTO
360 GOTO 140:REM RICOMINCIA
```

rapidità di comprensione.

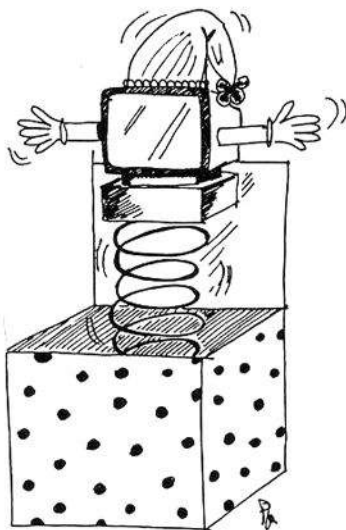
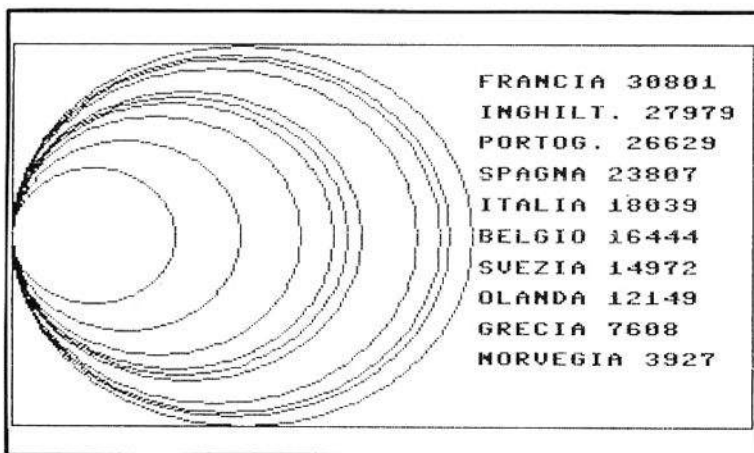
Se, però, ci accontentiamo di effetti meno... spettacolari, possiamo utilizzare la tecnica del programa n.2 che visualizza tre istogrammi parzialmente sovrapposti tra loro; quello centrale è disegnato "pieno" per facilitarne la lettura.

Per come è scritto il programma è possibile visualizzare solo tre isto-

```
100 REM AREE PROPORZIONALI
110 REM C/128, C/16, PLUS/4
120 :
130 PRINT CHR$(147):REM CANCELLA SCHERMO
140 PRINT"ECCO 10 VALORI CASUALI":PRINT
150 FOR I=1 TO 10:REM GENERA 10 VALORI RND
160 GOSUB 320:IF X<1000 THEN 160:REM SCARTA
170 X(I)=SQR(X/π):READ A$:REM LEGGE DATA
180 X$(I)=A$:PRINT X,X$(I): NEXT
190 PRINT: PRINT "PREMI UN TASTO..."
200 GETKEY A$: PRINT "ATTENDI..."
210 :
220 FOR J=1 TO 10: MA=X(J):REM INIZIO SORT
230 MA$=X$(J): FOR I=1 TO 10
240 IF X(I)=>MA THEN MA=X(I):MA$=X$(I):X=I
250 NEXT I:Y(J)=MA:Y$(J)=MA$:X(X)=0:NEXT J
260 GRAPHIC 1,1:REM INIZIA PARTE GRAFICA
270 FOR I=1 TO 10: CIRCLE 1,Y(I),100,Y(I)
280 W$=Y$(I)+STR$(INT(Y(I)*Y(I)*π))
290 CHAR 1,25,I*2,W$:REM VISUALIZZA SCRITTE
300 NEXT: GETKEY A$:GRAPHIC 0: END
310 REM 31415= AREA MASSIMA VISUALIZZABILE
320 X=INT(RND(0)*31415):RETURN
330 DATA ITALIA, SPAGNA, SVEZIA, FRANCIA
340 DATA OLANDA, BELGIO, GRECIA, INGHILT.
350 DATA PORTOG., NORVEGIA
```

grammi sovrapposti (o affiancati) alla volta, ma l'attento lettore sarà sicuramente in grado di apportare le dovute modifiche in modo da tracciare il numero di barre desiderate.

La prima domanda posta ("Ampiezza?") si riferisce alla larghezza di ciascun istogramma; naturalmente maggiore sarà tale valore, minore sarà il numero di barre visualizzabili in ciascuna schermata; in caso di valori troppo piccoli, al contrario, si potrebbero presentare problemi di leggibilità.

La seconda domanda, relativa allo sfalsamento, chiede il numero di pixel (puntini elementari) desiderati tra un istogramma e l'altro. Se tale valore è minore di AM, gli istogrammi risulteranno sovrapposti; se, invece, SF sarà eguale o maggiore di AM, gli istogrammi appariranno affiancati, più o meno vicini tra loro, realizzando un altro tipo di rappresentazione, peraltro nota.

La variabile XA(1) indicherà la distanza dal bordo sinistro del primo dei tre istogrammi; la distanza degli altri due viene determinata automaticamente. Le tre ordinate richieste, ovviamente, rappresenteranno la dimensione verticale delle corrispondenti barre.

La tecnica di tracciamento dei tre istogrammi sembra complessa, ma garantisce l'eliminazione di linee ed aree nascoste: la riga 280 provvede a tracciare il primo rettangolo "vuoto"; la 300 disegna il secondo "pieno". Se, a questo punto, tracciassimo il terzo rettangolo, questo verrebbe occultato, parzialmente, dal secondo; per evitare tale inconve-

NEGOZIO AL PUBBLICO
E VENDITA PER CORRISPONDENZA
CASH & CARRY

COMMODORE POINT '88

COMMODORE AMIGA 500/2000
"AL PREZZO PIÙ BASSO D'ITALIA" »CON GARANZIA & OMAGGIO«

Amiga VID L. 139.000
Digitalizzatore di immagini per Amiga 500/1000/200

Amiga Syntetic L. 170.000
Digitalizzatore Audio per Amiga 500/1000/2000

Amiga VIDEOSOUND L. 290.000
Digitalizzatore Audio VIDEO, tutto in uno.
Ottimo, per 500/1000

INT MIDI AMIGA PLUS L. 79.000
Nuova interfaccia midi per Amiga 500/1000/2000

DRIVE AGGIUNTIVO AMIGA 500/1000 (SLIM LINE
L. 235.000

ESPANSIONE 512 PER AMIGA 500
INT. con orologio L. 199.000

ESPANSIONE 2MB per A500/1000
L. 850.000

EMULATORE 64 per AMIGA L. 49.000
L'UNICO EMULATORE VERAMENTE FUNZIONANTE CON L'AUDIO E CON LA POSSIBILITÀ DI SALVARE I FILES SU 3 1/2

VID-AMIGA II L. 899.000
Novità digitalizzatore in tempo reale per A 500/1000/2000

NOVITÀ **PENNA OTTICA PER AMIGA 500/1000/2000** L. 149.000

NOVITÀ **DRIVE 5 1/4 per amiga** L. 299.000
Permette di CARICARE pvg. del PC-IBM

DRIVE INTERNO PER AMIGA 2000 (MECC. NEC)
L. 199.000

NOVITÀ **VIDEON (AMIGA)** L. 290.000
Nuovo digitalizzatore di immagini - Lavora senza filtri e in "PIÙ direttamente da Videoregistratore o Telecamera!

OLTRE 1000 PROGRAMMI AMIGA
RICHIEDERE CATALOGO

COMMODORE 64/128 - COMMODORE 64/128

THE NEW FINAL TURBO III per 64/128 (modo 64) L. 69.000
L'emulazione continua!!! Eccovi l'ultima release della mitica cartuccia notevolmente migliorata e modificata. Turbola favolosa routine dello speddos su cartuccia fino a 10 volte più veloce sia in lettura che in scrittura!!! 8 tasti funzione programmati. 24 K ram per i grog. in Basic. Un favoloso protettore di programmi tipc O.M.A. incorporati. Dischi e cassette IN UN SOLO FILE!!! (boot se necessita. Inoltre ha incorporato il GAME KILLER (evita la collisione degli sprite ed ha ben 40 comandi Basic Turbo a disposizione ... HARDCOPY "HP". Premendo un solo tasto potrete fare l'ardcopy del video in 12 gradazioni di grigio.
ECCEZIONALE!!!

EPRON NEW GRAPHIC MPS 803
Si sotituisce il generatore di caratteri della stampante MPS-80 (per migliorare la leggibilità della scrittura con quattro nuovi set di caratteri.
L. 49.000

PROCESSORE VOCALE
L. 115.000
Digitalizzatore vocale tipo "Voice Master" notevolmente migliorato composto a cartuccia hardware e microfono software interamente in italiano con ampio manuale di istruzioni. Incredibile, fa parlare il tuo Commcdore 64 puoi programmarlo a fin che conosca la tua voce e ti risponda.

KIT PULIZIA DRIVE
5 1/4 o 3 1/2
L. 15.000 cad.

DISPONIBILI TUTTI I PEZZI DI RICAMBIO CCOMMODORE 64
Sconti particolari per rivenditori e quantitativi.
TELEFONATE! Per ulteriori informazioni richiedete i cataloghi per il vostro Computer il settore, inviando L. 1.000 in francobolli. Ricorda che alla **NEWEL**trovi anche tutto per il COMMODORE AMIGA 64-128 MSX, SINCLAIR ZX & QL ATARI ST e PC compatibili.

NUOVA GUIDA AL CP/M 128 IN ITALIANO
L. 30.000

Alimentatori professionali
- - - -
per 64 L. 39.000
per 128 L. 69.000

VIDEODIGITAL 64
Nuovo digitalizzatore in cartuccia, digitalizza le tue più belle immagini con l'aiuto di una videocamera o videoregistratore semplicissimo da usare con manuale in italiano. Inoltre è possibile modificare le immagini con il KOALA ecc.
L. 70.000

STARDOS NEW! Eccezionale novità un velocizzatore che supera persino la velocità dello speed-dos attiva i tasti funzione ecc. In una sola Eprom kit da inserirsi nel C64 con manuale in ital. Non necessita di elaborazioni al drive ne del cavo parallelo.
L. 39.000

Adattatore telematico 64 Commodore V21-V23
TUTTO IN ITALIANO **L. 99.000**

REALTIME DIGITAL 64 **L. 170.000**
Notevolmente migliorato con possibilità di animazioni e videoclip.
IL PRIMO DIGITALIZZATORE
IN TEMPO REALE PER IL TUO 64/128

DRIVE OC-118 per 64/128
Drive Slim-line 100% compatibile
a sole L. 249.000

NOVITÀ **The CARTRIDGE (NOVITÀ) L. 79.000**
Nuova cartuccia multiutility con:
Turbo, Superturbo fino a 10 volte più veloce,
Sprite Killer, Poker, Monitor
Utility e comandi aggiuntivi, Toolkit,
tasti funzione, copyfiles, ecc.

PIÙ UN ECCEZIONALE SPROTETTORE
CASSETTA - DISCO E VICEVERSA
IL TUTTO IN UN UNICO FILE
TUTTO IN ITALIANO !!!
Compatibile speed-dos e drive compatibili!

NOVITÀ
NOVITÀ ASSOLUTA!!!
The ICONE CARTRIDGE
per 64/128
L. 90.000
Un nuovo sistema operativo a ICON e finestre grafiche che racchiudono le caratteristiche della final III & The cartridge 64 e molto più!! Potrete valorare quasi come con il Geos 64 (MAN. IN ITALIANO)

IN OFFERTA

O.M.A. PLUS	L. 60.000
TURBO FREEZE	L. 50.000
SPEED-DOS 1541	L. 49.000
SPEED-DOS 1541/C	L. 59.000
PENNA OTTICA	L. 49.000
TAST LOAD - RESET	L. 30.000
KIT EPROM 80	L. 25.000
KIT GRAFICO 802	L. 35.000
RESET 64	L. 10.000
COPRITASTIERA 64	L. 10.000
COPRITASTIERA 64 NEW	L. 15.000
COPRITASTIERA 128	L. 18.000

COMUNICATO

LA NEWEL ANNUNCIA UNA SVENDITA TOTALE

per rinnovo dei locali
tutto il materiale (compresi gli arredi),
sarà venduto con sconti reali dal 20% al 70%

ECCO ALCUNI ESEMPI

– CP/M EMULATOR PER C64	199.000	**39.000**
– CALC RESULT	199.000	**29.000**
– LOGO 64	175.000	**39.000**
– INTRODUZIONE AL BASIC I		**19.000**
– INTRODUZIONE AL BASIC II		**19.000**
– MAGIC DESK (Cartri ge)		**19.000**
– MODEM DIRETTO C-64 (con software)		**85.000**
– DISCHI VERGINI 51/4 DSDD BULK		**1.000**

(minimo 50 pezzi))

e tanti altri accessori per
C-64, AMIGA, e PC COMPATIBILI

PASSA A TROVARCI

NEWEL - Via Mac Mahon, 75 - Tel. 02/323492
NEVEL-BBS - Tel. 02/3270226 - 300/1200 band

niente si provvede dapprima a cancellarlo e, in seguito, a tracciarne i soli contorni (disegnandolo, cioè, vuoto).

E' facile accorgersi che, indipendentemente dai valori dei tre istogrammi, questi risulteranno sempre "sovrapposti" tra loro.

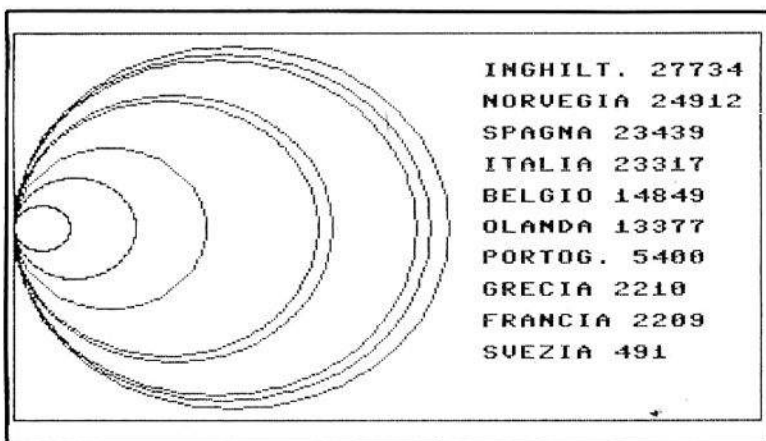
Il listato n.4 non è altro che il listato n.2, scritto, però, per girare sul C/64 dotato di emulatore Gw-Basic oppure su qualunque computer Ms-Dos compatibile.

Le aree proporzionali

A volte gli istogrammi non consentono di visualizzare chiaramente la differenza presente tra diverse grandezze, specialmente se queste sono assimilabili, con una certa fantasia, a grandezze bidimensionali, come le superfici.

Un istogramma, benchè sia certamente più facilmente comprensibile di un valore numerico, è poco indicato per confrontare tra loro grandezze relative a più di una dimensione; per queste ultime sarebbero più indicati istogrammi "superficiali" rappresentati, quindi, da aree.

Di solito la figura geometrica scelta per tale rappresentazione è il cerchio, meno frequentemente il quadrato.

Il programma n.3 genera 10 valori casuali compresi tra 1000 (per evitare valori altrimenti poco visibili) e 31415: quest'ultimo valore, infatti, rappresenta la superficie massima di un cerchio rappresentabile sullo schermo di un computer di dimensioni 40x25.

Sappiamo, infatti, che l'area di un cerchio è data dalla relazione:

A = r * r * 3.14

Siccome lo schermo è alto 200 pixel, ne consegue che il raggio del cerchio più grande (rappresentabile per intero) vale 100 pixel. Di conseguenza si ha:

100 * 100 * 3.14 = 31400

Se, poi, invece, di 3.14 usiamo 3.1415... si può comprendere l'approssimazione indicata in riga 310.

Stavamo prima dicendo che il programma associa a ciascuno dei 10 numeri generati casualmente, i 10 nomi di nazioni contenuti nei Data.

Subito dopo mette in ordine crescente (operazione detta, in gergo, "Sort") i 10 valori utilizzando due vettori di appoggio: Y() e Y$(). Al termine delle operazioni i due vettori conterranno, ripettivamente, i valori numerici ed i nomi corrispondenti.

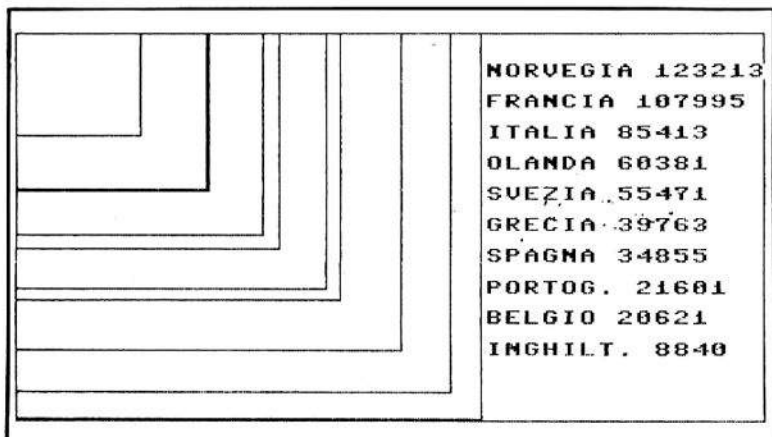
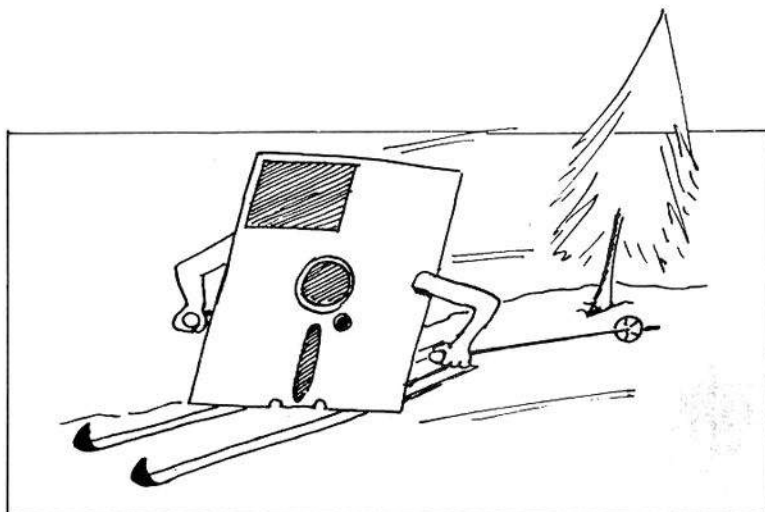
La tecnica di visualizzazione è piuttosto semplice: le 10 circonferenze vengono tracciate una alla volta a partire dalla più grande; contemporaneamente appaiono, alla destra del video, i dieci nomi seguiti dai valori numerici loro associati.

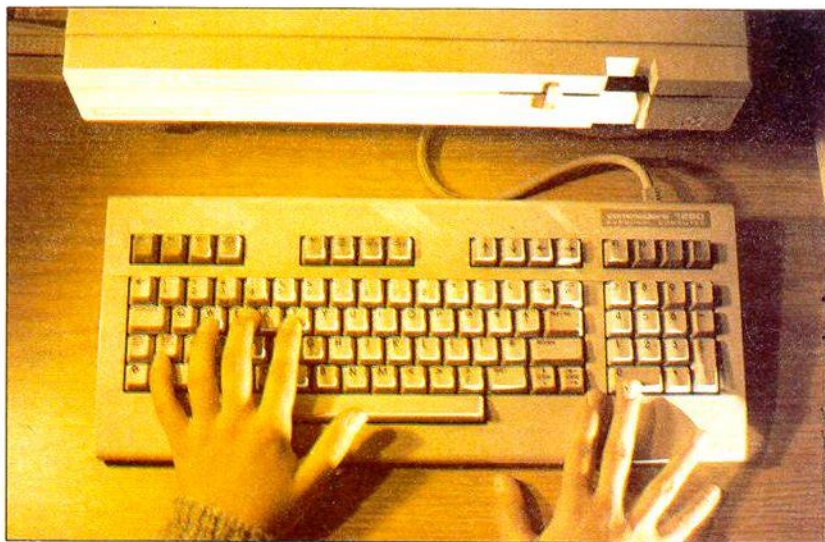
Può capitare il caso in cui due valori siano molto simili tra loro, se non addirittura uguali: in questo caso, purtroppo, le due circonferenze tracciate risultano eguali, pur se tracciate due volte, e sembra che tra le aree indicate ne manchi qualcuna.

Con un po' di pratica, tuttavia, si riconosce subito quale circonferenza "doppia" rappresenta, in realtà, due valori.

L'ultima figura si riferisce ad un altro programma (non pubblicato) che segue, praticamente, lo stesso concetto del precedente ma visualizza quadrati anzichè circonferenze.

Pur se, in teoria, le aree geometri-

che continuano ad essere in proporzione ai valori assegnati, non si può negare che la rappresentazione "circolare" è senza dubbio più comprensibile.

Un'ultima nota, prima di concludere: i cerchi, sul video, sono perfettamente... circolari. Nel passare dallo schermo alla carta, purtroppo, le dimensioni vengono alterate a causa di problemi tecnici connessi con la stampante stessa: di conseguenza vengono stampate ellissi invece di cerchi.

```
100 REM ISTOGRAMMI SOVRAPPOSTI
110 REM C/64 + GW-BASIC (QUALSIASI VERSIONE)
120 CLS 0:REM CANCELLA SCHERMO
140 SCREEN 0:REM TESTO
150 INPUT"AMPIEZZA";AM:REM LARGH.ISTOGRAMMA
160 INPUT"SFALSAM.";SF:REM SFALSAMENTO ISTOG
170 INPUT"ASCISSA 1";XA(1):REM POSIZIONE
180 REM POSIZIONAM. AUTOMATICO DELLE ASCISSE
190 XB(1)=XA(1)+AM:REM RIFERIMENTO PRIMARIO
200 XA(2)=XA(1)+SF:XB(2)=XA(2)+AM
210 XA(3)=XA(2)+SF:XB(3)=XA(3)+AM
220 REM INPUT DELLE ORDINATE CORRISPONDENTI
230 INPUT"ORDINATA 1";YA(1)
240 INPUT"ORDINATA 2";YA(2)
250 INPUT"ORDINATA 3";YA(3)
260 SCREEN 1:CLS 1:REM MODO GRAFICO HI-RES
270 REM PRIMO ISTOGRAMMA VUOTO
280 LINE (XA(1),YA(1))-(XB(1),150),1,B
290 REM SECONDO ISTOGRAMMA "PIENO"
300 LINE (XA(2),YA(2))-(XB(2),150),1,BF
310 REM TERZO ISTOGRAMMA "CANCELLATO"
320 LINE (XA(3),YA(3))-(XB(3),150),0,BF
330 REM TERZO ISTOGRAMMA VUOTO
340 LINE (XA(3),YA(3))-(XB(3),150),1,B
350 A$=INKEY$:REM ATTESA PRESSIONE TASTO
355 IF A$="" THEN 350
360 GOTO 140:REM RICOMINCIA
```

DALLO SCHERMO ALLA STAMPANTE

Come inviare su stampante la schermata in bassa risoluzione del C/128 in modo 40 oppure 80 colonne

Il programma di queste pagine consente di ottenere la copia su carta dello schermo a 40 oppure 80 colonne alla velocità tipica del linguaggio macchina.

E' molto semplice da usare perchè basta battere...

SYS 5120, S, M

...in cui "S" seleziona lo schermo: 0 per quello ad 80 colonne, ed un qualsiasi altro valore per quello a 40 colonne.

Il valore di "M" ordina al computer di stampare in minuscolo (se posto a zero) o maiuscolo (qualsiasi altro valore).

Il programma utilizza i codici della MPS-803; funzionerà, quindi, per qualsiasi stampante 803-compatibile.

La routine è facilmente rilocabile, contenendo soltanto tre salti assoluti. Si è fatto largo uso delle routine del S.O., tra cui:

$FFC3: Chiude un file precedentemente aperto. Richiede l'inserimento in Accumulatore del numero del file da chiudere.

$FFBA: Setta il numero di identificazione del file, il numero del dispositivo e del comando; da mettere, rispettivamente, in A, X, Y.

$FFC0: Apre il file precedentemente indicato con $FFBA.

$FFC9: Predispone un file per l'output. Necessita del numero del file in X.

$FFD2: Manda il carattere al device di output. Il codice ASCII del carattere va posto in Accumulatore.

$FFCC: Resetta i canali di I/O.

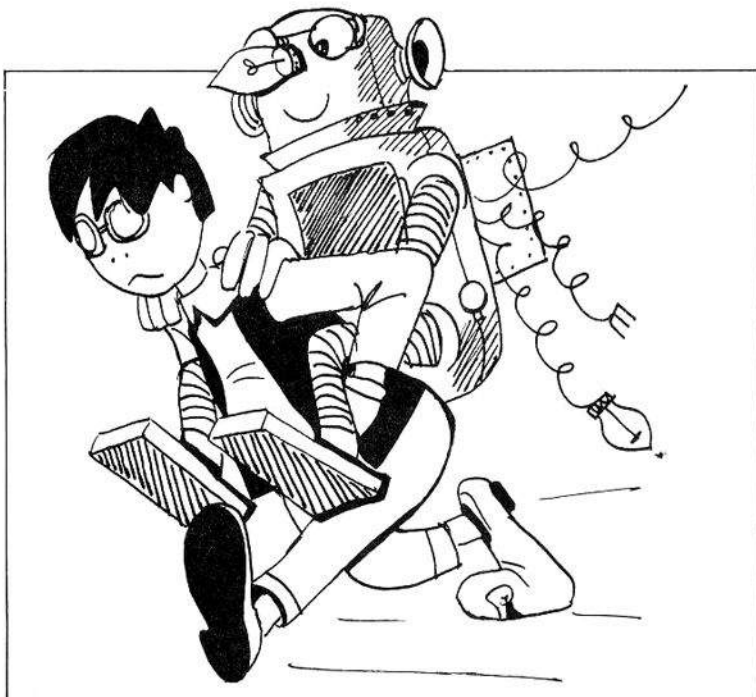
$CDCC: Effettua una POKE nella memoria dell'8563; necessita del numero del registro in X, e del valore da assegnare in A.

$CDDC: Effettua una PEEK nella memoria dell'8563; il numero del registro da leggere va in X, mentre il suo valore si troverà in A.

Tutti i simboli di doppi apici (") saranno automaticamente trasformati in singoli (') per evitare inconvenienti di vario tipo. Il programma è in grado di accorgersi se un carattere è in negativo e di stamparlo come tale.

```
  Routine Hard Copy
in bassa risoluzione per
C/128 in modo 40/80 col.

. 01400  85 fb     sta $fb
. 01402  a9 91     lda #$91
. 01404  85 ff     sta $ff
. 01406  8a        txa
. 01407  f0 04     beq $140d
. 01409  a9 11     lda #$11
. 0140b  85 ff     sta $ff
. 0140d  ad 00 ff  lda $ff00
. 01410  85 fc     sta $fc
. 01412  a9 00     lda #$00
. 01414  8d 00 ff  sta $ff00
. 01417  a9 20     lda #$20
. 01419  20 c3 ff  jsr $ffc3
. 0141c  a9 20     lda #$20
. 0141e  a2 04     ldx #$04
. 01420  a0 0a     ldy #$0a
. 01422  20 ba ff  jsr $ffba
. 01425  a9 20     lda #$20
. 01427  20 c3 ff  jsr $ffc3
. 0142a  a9 20     lda #$20
. 0142c  a2 04     ldx #$04
. 0142e  a0 ff     ldy #$ff
. 01430  20 ba ff  jsr $ffba
. 01433  20 c0 ff  jsr $ffc0
. 01436  a2 20     ldx #$20
. 01438  20 c9 ff  jsr $ffc9
. 0143b  a5 ff     lda $ff
. 0143d  20 d2 ff  jsr $ffd2
. 01440  a5 fb     lda $fb
. 01442  f0 0f     beq $1453
. 01444  a9 00     lda #$00
. 01446  85 fd     sta $fd
. 01448  a9 04     lda #$04
. 0144a  85 fe     sta $fe
. 0144c  a9 00     lda #$00
. 0144e  85 fa     sta $fa
. 01450  18        clc
. 01451  90 12     bcc $1465
. 01453  a9 00     lda #$00
. 01455  a2 12     ldx #$12
. 01457  20 cc cd  jsr $cdcc
. 0145a  a2 13     ldx #$13
. 0145c  20 cc cd  jsr $cdcc
. 0145f  20 d8 cd  jsr $cdd8
. 01462  18        clc
. 01463  90 04     bcc $1469
. 01465  a0 00     ldy #$00
. 01467  b1 fd     lda ($fd),y
. 01469  c9 80     cmp #$80
. 0146b  30 09     bmi $1476
. 0146d  29 7f     and #$7f
. 0146f  48        pha
. 01470  a9 12     lda #$12
. 01472  20 d2 ff  jsr $ffd2
. 01475  68        pla
. 01476  c9 22     cmp #$22
. 01478  d0 05     bne $147f
. 0147a  a9 27     lda #$27
. 0147c  18        clc
. 0147d  90 1b     bcc $149a
. 0147f  c9 20     cmp #$20
. 01481  10 05     bpl $1488
. 01483  69 40     adc #$40
. 01485  18        clc
. 01486  90 12     bcc $149a
. 01488  c9 40     cmp #$40
. 0148a  10 03     bpl $148f
. 0148c  18        clc
. 0148d  90 0b     bcc $149a
. 0148f  c9 60     cmp #$60
. 01491  10 05     bpl $1498
. 01493  69 20     adc #$20
. 01495  18        clc
. 01496  90 02     bcc $149a
. 01498  69 40     adc #$40
. 0149a  20 d2 ff  jsr $ffd2
. 0149d  a9 92     lda #$92
. 0149f  20 d2 ff  jsr $ffd2
. 014a2  a5 fb     lda $fb
. 014a4  f0 36     beq $14dc
. 014a6  e6 fa     inc $fa
. 014a8  a5 fa     lda $fa
. 014aa  c9 28     cmp #$28
. 014ac  d0 09     bne $14b7
. 014ae  a9 0d     lda #$0d
. 014b0  20 d2 ff  jsr $ffd2
```

```
. 014b3  a9 00     lda #$00
. 014b5  85 fa     sta $fa
. 014b7  e6 fd     inc $fd
. 014b9  d0 02     bne $14bd
. 014bb  e6 fe     inc $fe
. 014bd  a5 fe     lda $fe
. 014bf  c9 07     cmp #$07
. 014c1  d0 a2     bne $1465
. 014c3  a5 fd     lda $fd
. 014c5  c9 e7     cmp #$e7
..014c7  d0 9c     bne $1465
. 014c9  a9 0d     lda #$0d
. 014cb  20 d2 ff  jsr $ffd2
. 014ce  a9 20     lda #$20
. 014d0  20 c3 ff  jsr $ffc3
. 014d3  20 cc ff  jsr $ffcc
. 014d6  a5 fc     lda $fc
. 014d8  8d 00 ff  sta $ff00
. 014db  60        rts
. 014dc  a2 12     ldx #$12
. 014de  20 da cd  jsr $cdda
. 014e1  c9 07     cmp #$07
. 014e3  f0 03     beq $14e8
. 014e5  4c 5f 14  jmp $145f
. 014e8  a2 13     ldx #$13
. 014ea  20 da cd  jsr $cdda
. 014ed  c9 d0     cmp #$d0
. 014ef  f0 03     beq $14f4
. 014f1  4c 5f 14  jmp $145f
. 014f4  4c c9 14  jmp $14c9
```

```
100 REM****************************
110 REM** HARD COPY C/128  ********
120 REM** NOTE PER L'USO : ********
130 REM** HARDCOPY 80 COLONNE *****
140 REM** SYS 5120,0,0 (MAIUSCOLO)
150 REM** SYS 5120,0,1 (MINUSCOLO)
160 REM** HARDCOPY 40 COLONNE *****
170 REM** SYS 5120,1,0 (MAIUSCOLO)
180 REM** SYS 5120,1,1 (MINUSCOLO)
190 REM****************************
200 :
210 IN=5120:CO=6:B=3:FAST
220 DO
230 READ A:IF A=-1 THEN EXIT
240 POKE IN,A
250 CO=CO+B:B=-B
260 CK=CK+A:IN=IN+1
270 LOOP
280 :
290 SLOW: READ CH:SCNCLR
300 IF CH<>CK THEN 720
310 END
320 :
330 DATA 133,251,169,145,133
340 DATA 255,138,240
350 DATA 4,169,17,133,255,173,0,255
360 DATA 133,252,169,0,141,0,255,169
370 DATA 32,32,195,255,169,32,162,4
380 DATA 160,10,32,186,255,169,32,32
390 DATA 195,255,169,32,162,4,160,255
400 DATA 32,186,255,32,192,255,162,32
410 DATA 32,201,255,165
420 DATA 255,32,210,255
430 DATA 165,251,240,15,169,0,133,253
440 DATA 169,4,133,254,169,0,133,250
450 DATA 24,144,18,169,0,162,18,32
460 DATA 204,205,162,19,32,204,205,32
470 DATA 216,205,24,144,4,160,0,177
480 DATA 253,201,128,48,9,41,127,72
490 DATA 169,18,32,210,255,104,201,34
500 DATA 208,5,169,39,24,144,27,201
510 DATA 32,16,5,105,64,24,144,18
520 DATA 201,64,16,3,24,144,11,201
530 DATA 96,16,5,105,32,24,144,2
540 DATA 105,64,32,210,255,169,146,32
550 DATA 210,255,165,251,240
560 DATA 54,230,250
570 DATA 165,250,201,40,208,9,169,13
580 DATA 32,210,255,169,0,133,250,230
590 DATA 253,208,2,230,254
600 DATA 165,254,201
610 DATA 7,208,162,165,253
620 DATA 201,231,208
630 DATA 156,169,13,32,210,255,169,32
640 DATA 32,195,255,32,204
650 DATA 255,165,252
660 DATA 141,0,255,96,162,18,32,218
670 DATA 205,201,7,240,3,76,95,20
680 DATA 162,19,32,218,205
690 DATA 201,208,240
700 DATA 3,76,95,20,76,201,20
710 DATA -1,32217
720 PRINT"ERRORE NEI DATA":END
```

Il manuale del Commodore 16

La traduzione in lingua italiana della guida all'utente per il C/16 non riporta l'indice alfabetico degli argomenti trattati nel volume.

Ne consegue, spesso, una ricerca difficoltosa dal momento che le varie istruzioni, comandi e funzioni non si succedono secondo un ordine facilmente individuabile dall'utente.

Certi di fare cosa gradita ai possessori del piccolo computer, riportiamo, qui di seguito, l'elenco alfabetico completo dei numerosi argomenti e delle pagine relative in cui sono pubblicati.

L'indice riportato all'inizio del manuale allegato al computer, invece, è sufficiente per individuare i capitoli relativi agli argomenti trattati.

Abbrev. coman. 127
Abs 117
Alta risoluzione 63
Animaz. caratteri 58
Argomento (def.) 81
Asc 117
Ascii (cod.) 144
Atn 117
Auto 82
Backup 82
Box 66, 91
Caratteri grafici 18, 54
Caricare 23, 24, 26
Cartridges 22
Cassette 23
Cerchio 93
Char 65, 92
Chr$ 121, 144
Circle 68, 93
Close 94
Clr 94
Clr/Home 15
Cmd 94
Collect 83
Color 60, 95
Colore (tasti) 17, 23
Comandi 80
Commodore (tasto) 15
Cont 83
Copy 83
Correzione err. 36
Cos 117
Ctrl 15
Data 95
Dec 117
Def Fn 95
Del 37
Delete 84
Dim 96
Directory 29, 84
Dischetti 25
Dload 85
Do 72, 97

Draw 64, 97, 117
Dsave 86
Effetti sonori 74
Ellisse 93
Else 101
End 97
Err$ 121
Errori (mess.) 131
Errori (Dos) 134
Esc 17, 41
Espressione (def.) 82
Exit 97
Exp 118
Finestre 41
Flash (on/off) 16
Fn 118
For 98
Frazioni 45
Fre 122
Frequenza 73
Funz. numeriche 51
Funzione (def.) 80
Funzione (tasti) 18
Funz. matem. 139
Get 99
Getkey 99
Gosub 100
Goto 100
Grafici (tasti) 18, 54
Graphic 62, 70, 100
Graphic clr 62, 101
Gshape 114
Header 27, 86
Help 86
Help (tasto) 19
Hex$ 121
If 101
Input 102
Inst/Del 14
Instr 118
Int 118, 120
Istruzione (def.) 80
Joy 118

Joystick 3
Key 87
Left$ 121
Len 121
Let 103
List 87
Load 88
Locate 64, 103, 117
Log 119
Loop 72, 97
Mid$ 121
Monitor 104
Multicolor 70
Musica 75, 141
Next 98, 104
New 88
Notaz. scient. 45
Note music. 141
On 104
Open 105
Operatori 80, 125
Operazioni 44
Ordine di calcolo 47
Ottagono 93
Paint 69, 106
Parametro (def.) 81
Peek 119
Pigreca 122
Poke 107
Poligoni 68
Pos 122
Print 107, 108
Print using 108
Pudef 111
Rclr 119
Rdot 103, 119
Read 111
Rename 89
Renumber 89
Reset 3
Restore 112
Resume 112
Return (istr.) 112

Return (tasto) 13
Rgr 119
Right$ 122
Rlum 119
Rnd 119
Rumori 75
Run 89
Run/Stop 13
Rvs (on/off) 16, 33
Salvare 24, 28
Save 90
Scale 113
Scnclr 113
Scratch 90
Sgn 120
Shift 13
Sin 120
Sqr 120
Sound 72, 113
Spc 122
Sshape 114
Step 98
Stop 115
Str$ 122
Sys 115
Tab 122
Tan 120
Then 101
Ti$ 124
To 98
Trap 115
Triangolo 93
Troff 116
Tron 116
Until 72, 97
Usr 120
Val 120
Variabili 49, 80, 123
Verify 91
Vol 72, 116
Wait 116
While 97

Come leggere le recensioni

Ogni mese, su queste pagine, verranno esaminati e testati i videogame più recenti per i computer Commodore 64 ed Amiga.

Ad ogni descrizione verranno associate una immagine, catturata tra le più belle schermate, ed una breve pagella.

Quest'ultima, pur se, inevitabilmente, frutto di impressioni personali di chi esamina il gioco stesso, ha lo scopo di assegnare una valutazione del livello del software,soprattutto tenendo conto di altri game analoghi disponibili sul mercato.

La pagella, comprende cinque voci:

IMPATTO: **indica il livello di interesse suscitato dalla presentazione e dal tema del gioco.**

SCENARIO: **riguarda l'accuratezza con cui è realizzata la grafica e l'efficacia dei disegni degli sprite.**

SUONO: **valuta gli effetti sonori presenti e le eventuali musiche di sottofondo.**

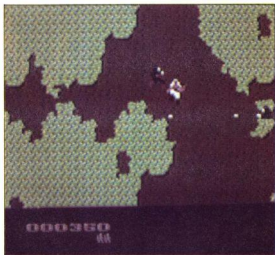
INTERESSE: **si riferisce al livello di interesse che il gioco può suscitare in un giocatore abituato ai videogame e, implicitamente, alla sua probabilità di "permanenza" sui vostri monitor.**

TOTALE: **ha lo scopo di sintetizzare i precedenti valori con un voto unico.**

HAWK MISSION C/64-128

La dura guerra iniziata da tempo immemorabile, combattuta tra il tuo paese e le malvagie forze di Turkal, sta per portare all'estinzione completa del tuo popolo, e nessuna delle due fazioni sembra decisa a retrocedere.

Tu stesso, dopo molti sforzi, ti sei arreso all'evidenza dei fatti: la grande rivalità che domina i leader degli stati in guerra è talmente radicata che ogni tentativo di riappacificare civilmente i due paesi rischia di risolversi in un vero e proprio suicidio.

Vedendo, però, che l'evolversi degli eventi sembra aver condannato la tua patria a sconfitta sicura, ed essendo tu un grande guerriero, allenato dal maestro Lim, proponi al Grande Turkal, capo dei tuoi nemici e amante del rischio, un'azzardata soluzione: se da solo, armato unicamente di coltelli, riuscirai a sconfiggere un'intera armata di cavalieri turkaliani, sopravvivendo alle insidie del deserto del Kyr e del canale di Yldiz, Turkal dichiarerà apertamente la propria sconfitta.

E allora, vai! Non indugiare, perchè solo tu puoi salvare il popolo da una morte sicura.

HAWK MISSION è presente, assieme a tanti altri giochi sul disco Commodore 64 Club N.5.

MIDNIGHT C/64-128

Attenzione... è quasi mezzanotte!

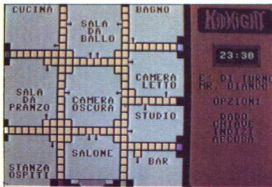
Quando scocca questa fatidica ora, possono accadere incredibili storie di straordinaria follia.

In una villa, situata ai confini della realtà, sta per compiersi un atroce delitto.

Ti trovi in questo macabro luogo per impedire, a chi si nasconde in agguato dietro qualche angolo buio, la riuscita del piano diabolico creato da una mente criminale.

Chi è la vittima dell'imprevedibile pazzo che s'annida nella villa?

Dove agirà e quale sarà l'arma del delitto?

Un bellissimo gioco di strategia ispirato al famoso CLUEDO presente sul numero 5 del disco Commodore 64 Club.

AMADEUS' REVENGE C/64-128

Siete il famoso Wolfang Amadeus Mozart che, in pochi minuti prima del concerto, deve assolutamente eliminare, dalla sua partitura, le note stonate che l'invidioso maestro Salieri ha diabolicamente introdotto in precedenza.

E' una specie di "incubo" musicale che vi vedrà impegnati nell'impersonare Wolfang Amadeus Mozart in un videogame coloratissimo e pieno di effetti sonori.

La grafica ed il sonoro sono molto ben curati e coinvolgono il giocatore.

Amadeus' Revenge fa parte del disco Commodore 64 Club N.5

PLATOON
C/64-128

Da un film di successo un grande videogame che appassionerà senz'altro tutti gli appassionati di questo genere.

E' diviso in più "sottogiochi" che impegnano il soldato/giocatore su diversi fronti di combattimento.

La grafica (specialmente nei movimenti) ed il sonoro (abilmente digitalizzato), fanno di questo gioco un ottimo prodotto.

IMPATTO:	*8*
SCENARIO:	*8*
SUONO:	*8*
INTERESSE:	*8*
TOTALE:	**8**

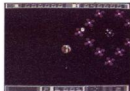
M.A.C.H.
C/64-128

Un nuovo shoot'em up ben curato dal punto di vista della grafica e del sonoro.

Il protagonista è una specie di "bio-computer" che, sorvolando una città (molto ben disegnata), combatte fino all'ultimo sangue con numerosi nemici volanti che cercano in tutti i modi di neutralizzarlo.

Buona la grafica e buono il sonoro; peccato che il tipo di gioco sia un po' troppo inflazionato.

IMPATTO:	*6*
SCENARIO:	*7*
SUONO:	*7*
INTERESSE:	*6*
TOTALE:	**6**

HUNTER'S MOON
C/64-128

C'è una graziosa astronave in giro per l'universo.

I nemici, come di consueto, sono numerosi e pericolosi ed il nostro eroe dovrà farne di fatica per sconfiggerli tutti.

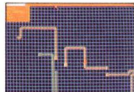
In sostanza si tratta di un giochino giocabile e divertente, ma per un periodo di tempo limitato.

IMPATTO:	*6*
SCENARIO:	*7*
SUONO:	*7*
INTERESSE:	*6*
TOTALE:	**6**

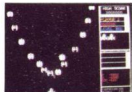
BLACK SHADOW
Amiga

Si tratta di un classico gioco a scorrimento verticale e tutto-grilletto o, come dicono gli americani, uno "scrolling zapper". Si può giocare, in due, anche a turno, usando i joystick od il mouse per controllare una piccola astronave che viaggia su di un pianeta dall'intensissimo traffico. Bisogna sparare a tutto. Grafica piccola, variegatissima, ma un po' lenta.

IMPATTO:	*7*
SCENARIO:	*7*
SUONO:	*5*
INTERESSE:	*5*
TOTALE:	**6**

TRACERS
Amiga

Chi non ricorda Tron, uno dei videogiochi più classici? Questa versione Amiga brilla per il numero elevatissimo di opzioni, che permettono di giocare da soli, o in due contemporaneamente, con varie modalità, schermi e perfino con varie musiche di accompagnamento. La grafica è di tipo semplicissimo, ma molto nitida e piuttosto veloce.

IMPATTO:	*7*
SCENARIO:	*5*
SUONO:	*8*
INTERESSE:	*6*
TOTALE:	**6**

VYPER
Amiga

Una versione Amiga del classico "Galaxians", ovvero un "Invaders" con gli alieni, se possibile, più agitati. La grafica è di piccole dimensioni, ma veloce ed animata con tecnica ineccepibile. Forse difettosa la rilevazione delle collisioni (a nostro vantaggio). Presente un numero incalcolabile di tipi di alieni, ciascuno con la sua modalità di movimento, sagoma e volume di fuoco.

IMPATTO:	*6*
SCENARIO:	*6*
SUONO:	*5*
INTERESSE:	*5*
TOTALE:	**5**

INSANITY FIGHT
Amiga

Si tratta di un programma nuovo, diverso da quello della Microstar già recensito: la grafica, infatti, è assai meno spettacolare ma, sul dischetto, è presente anche un Construction Set. Ovvero è presente un programma che rende possibile costruirsi gli scenari da sorvolare in scorrimento verticale con la nostra astronave ed i cannoni e gli alieni che dobbiamo affrontare. Idea vecchia, con grafica povera, ma gioco personalizzabile.

IMPATTO:	*6*
SCENARIO:	*5*
SUONO:	*5*
INTERESSE:	*6*
TOTALE:	**5**

THUNDERBOY
Amiga

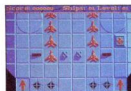
Un giochino del tipo "BC Quest for Tires". Si controlla un cavernicolo che deve recuperare la sua bella, correndo in scorrimento laterale, ad un solo senso, attraverso schermi pieni di avversità da evitare con salti o tempismo (pipistrelli, stalattiti, scorpioni...) ed, ovviamente, anche oggetti da collezionare. La grafica è piuttosto scarsa ed "a bassa risoluzione" per Amiga, (tipo C/64, o peggio) poco varia e colorata, anche se mossa molto bene.

IMPATTO:	*6*
SCENARIO:	*5*
SUONO:	*4*
INTERESSE:	*4*
TOTALE:	**5**

ARKANOID Amiga

Dopo tante versioni ufficiose (Amegas, Crystal Hammer...) finalmente l'originale videogioco dalla Taito, convertito per Amiga dalla Discovery. Dovete far rimblzare una pallina contro un muro per demolirlo completamente, con alcune varianti determinate dalla cattura di oggetti che si staccano dall'alto. Qui è possibile selezionare lo schermo dal quale iniziare.

IMPATTO:	*8*
SCENARIO:	*8*
SUONO:	*6*
INTERESSE:	*8*
TOTALE:	**8**

IRIDON Amiga

Uno "scrolling zapper" verticale, fratello povero dei vari Xenon e Black Shadow. La grafica è velocissima, sia nello scorrimento della piattaforma sottostante che nel movimento delle navi aliene, ma poco varia e ancor meno colorata. Un programma non nuovo, senza incentivi, da lasciare senza rimpianti nella vetrina del rivenditore.

IMPATTO:	*6*
SCENARIO:	*5*
SUONO:	*5*
INTERESSE:	*6*
TOTALE:	**5**

GROSSMEISTER CHESS Amiga

Un programma per giocare a scacchi scritto in Germania, utilizzante a fondo le capacità di memoria di Amiga (infatti bisogna disconnettere i drive supplementari perchè funzioni con 512 KRam). Sebbene le scritte dei menu a sipario siano in tedesco, chi ha già manovrato con programmi di questo genere non avrà problemi a capire quali sono le varie opzioni, tipiche ormai per tutti i programmi.

IMPATTO:	*7*
SCENARIO:	*5*
SUONO:	*6*
INTERESSE:	*6*
TOTALE:	**5**

JET Amiga

Una impressionante simulazione di volo con F16 e F18, probabilmente superiore al classico Flight Simulator (almeno, per giocosità, anche se tecnicamente meno preciso) della stessa SubLOGIC. E' possibile duellare in cielo, visionare dall'esterno, in tempo reale, il nostro aereo, cambiare scenografie (inserendo dischetti supplementari) sempre coloratissime e piene di particolari.

IMPATTO:	*9*
SCENARIO:	*8*
SUONO:	*7*
INTERESSE:	*8*
TOTALE:	**8**

JOE BLADE Amiga

Un divertente giochino con grafica tipo cartone animato. Si controlla un soldatino su grafica bidimensionale che deve penetrare in una base nazista a suon di colpi di mitra. Come in una avventura, è bene tracciare una mappa dei luoghi visitati, in quanto vi è un dedalo di porte e di oggetti da prendere (chiavi, uniformi nemiche, munizioni, armi...). Unica pecca della grafica è che il fucile non spara "graficamente", mentre gli effetti sonori e l'atmosfera sono ottimi.

IMPATTO:	*8*
SCENARIO:	*8*
SUONO:	*7*
INTERESSE:	*8*

INVADERCRAFT Amiga

Chi non conosce "Invaders", il primo videogioco a colori (PONG era in bianco e nero)?. Crediamo che implementare questo classico giochino con la base che spara cannonate verso alieni che scorrono a righe in alto sia difficile, perchè ogni computer ne ha dozzine di versioni. Questa per Amiga, una delle prime che vediamo, è molto ben fatta, con grafica di grandi dimensioni (OK anche per TV) e suoni eccellenti. Produzione tedesca.

IMPATTO:	*6*
SCENARIO:	*6*
SUONO:	*7*
INTERESSE:	*5*
TOTALE:	**6**

STARLAYS Amiga

E' il gioco della sfera che corre su strade in prospettiva, controllata dal joystick, e che deve saltare i buchi nel terreno. Qui, invece della sfera, abbiamo un simpatico omino, con scenografie sempre diverse ed animazione velocissima. Il numero di salti ed il tempo sono limitati. La grafica è di grosse dimensioni.

IMPATTO:	*6*
SCENARIO:	*6*
SUONO:	*5*
INTERESSE:	*6*
TOTALE:	**5**

SKY BLASTER Amiga

Una simpatica via di mezzo tra la simulazione di un elicottero ed un arcade tipo "tuttogrilletto". La visione è in prospettiva 3D molto rapida ed abbastanza rifinita, anche se i particolari non abbondano. Mancano tutte le cose e le opzioni che rendono complicatissimi i programmi di simulazione, ma così è più giocabile.

IMPATTO:	*7*
SCENARIO:	*7*
SUONO:	*6*
INTERESSE:	*7*
TOTALE:	**6**

QUANTO COSTA IL TUO COMMODORE

LISTINO PREZZI HOME COMPUTER

Codice	Prodotto	Prezzo (IVA esclusa)
	Personal Computer	
A 500	Personal Computer 16/32 BIT CPU 512K RAM incorpora un floppy disk drive da 3" 1/2 da 880Kb e un mouse	**L. 950.000**
A 501	Espansione di memoria per Amiga 500 512Kbytes e orologio	**L. 212.500**
A 520/1	Modulatore per Amiga 500 permette di collegare Amiga 500 ad un qualsiasi televisore b/n e colori	**L. 40.000**
A SCART	Cavo collegamento Amiga 500 con TV prescart	**L. 27.000**
Nuovo C64	Nuovo personal computer CPU 64K RAM computer ad alta risoluzione grafica, 256 combinazioni di colori, sintetizzatore di suono. Collegabile ad un qualsiasi televisore	**L. 375.000**
1764	Espansione di memoria per C64 256Kbytes	**L. 195.000**
C 128D	Personal computer CPU 128Kb RAM CPU 128Kbytes espandibile a 512Kbytes, 48Kbytes ROM, basic 7,0. Tastiera separata. Alta risoluzione grafica, 16 colori + 8 sprites. 40 80 (RGB) colonne. Programmabile in CP/M 3.0. Funzionante in modo C64. Floppy disk da 340Kb incorporato.	**L. 895.000**
1700	Espansione di memoria per C128. 128Kb	**L. 165.000**
1750	Espansione di memoria per C128. 512Kb	**L. 235.000**
	Registratori	
1530	Registratore compatibile C64 - C128 - C128D	**L. 55.000**
	Unità a dischi	
1541 II	Floppy disk drive 5" 1/4. Unità di memoria di massa, drive singolo, capacità 170Kbytes in linea. Compatibile con C64	**L. 395.000**
1581	Floppy disk drive 3" 1/2. Unità di memoria di massa, drive singolo da 3" 1/2, capacità 800Kbytes. Compatibile C64 - C128 - C128D	**L. 420.000**
A 1010	Floppy disk drive esterno 3" 1/2	**L. 395.000**
1802	Monitor a colori, alta risoluzione, 14" con audio. Collegabile al C64, C128, C128D (40 colonne)	
2080	Monitor a colori alta persistenza alta risoluzione, 14", con audio antiriflesso. Collegabile ad Amiga PC, C64, C128, C128D. 640x400 pixel, 4096 colori	**L. 630.000**
1084	Monitor a colori. Alta risoluzione, 14", con audio, antiriflesso. Collegabile al C64 - C128 - C128D - Amiga - PC. 640x400 pixel, 4096 colori.	**L. 545.000**
	Stampanti	
MPS 1200 S	Stampante. 80 colonne, 120 cps, bidirezionale, carta in modulo singolo e trascinamento modulo continuo per C64 - C128 - C128D	**L. 480.000**

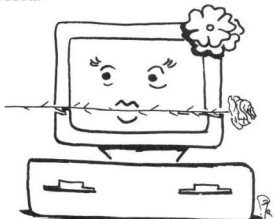
MPS 1200 P	Stampante 80 colonne 120 cps, bidirezionale, carta in modulo singolo e trascinamento modulo continuo per A500	**L. 480.000**
MPS 1250	Stampante con stesse caratteristiche del modello MPS 1200 P, interfacce seriale RS232C e parallela Centronics di serie	**L. 495.000**
MPS 1500	Stampante a colori. 80 colonne, 130 cps, 4 colori, bidirezionale, carta in modulo singolo e trascinamento modulo continuo.	**L. 550.000**
	Accessori	
6499	Adattatore telematico omologato. Collegabile al C64, permette il collegamento a Videotel PGE-Banche Dati.	**L. 149.000**
1351	Mouse. Per C64, C128, C128D	**L. 99.000**
1311	Joystick. Comando per giochi	**L. 13.500**
1399	Joystick. 8 microswitch-autofire	**L. 29.000**

LISTINO PREZZI SISTEMI

Codice		**Prezzo (IVA esclusa)**
PCI	Microprocessore 8088 (coprocessore 8087 opzionale) frequenza clock 4,77MHz. RAM 512Kb. Espandibile a 640Kb; 1 floppy da 360Kb. Monitor a fosfori verdi 12".	**L. 945.000**
PC 10-III	Microprocessore Intel 8088 a 16 bit; clock 10 MHz; memoria RAM: 640 Kb; memoria ROM (BIOS): 8 Kb autoconfigurabile; sistema operativo MS-DOS 3.21; GW-Basic; scheda video integrata monocromatico/colore di serie con emulazioni CGA/MDA/Hercules via software; 2 floppy disk drive da 360 Kb; monitor monocromatico a fosfori verdi da 12"; porte seriale RS232C e paralella Centronics; tastiera avanzata 102 tasti con 12 tasti funzione; porta mouse, incluso controller per H.D.	**L. 1.490.000**
PC 10-IIIC	Stessa configurazione ma con monitor 14" a colori mod. 1084	**L. 1.890.000**
PC 20-III	Microprocessore Intel 8088 a 16 bit; memoria RAM da 640Kb; memoria ROM (BIOS) da 8Kb autoconfigurabile. Sistema operativo MS-DOS 3.21. Scheda video monocromatico/colore di serie con emulazioni CGA/MDA/Hercules via software. Un hard disk da 20Mb e un floppy disk drive da 360Kb. Monitor monocromatico a fosfori verdi da 12". Porta seriale RS232C parallela Centronics. Porta mouse incluso controller per H.D.; tastiera avanzata 102 tasti con 12 tasti funzione	**L. 2.240.000**
PC 20-IIIC	Stessa configurazione ma con monitor 14" a colori mod. 1084	**L. 2.640.000**
PC 40/20 AT	Microprocessore 80286 a 16 bit; clock 10/6 MHz; memoria RAM dal 1 Mb; hard disk da 20 Mb; un floppy disk drive da 1,2 Mb; Sistema operativo MS-DOS 3.2, GW-Basic; scheda video monocromatico/colore 132 colonne AGA di serie; monitor monocromatico a fosfori verdi da 14"; porta seriale RS232C e parallela Centronics; tastiera avanzata 102 tasti con 12 tasti funzione	**L. 3.490.000**
PC 40/40 AT	Microprocessore 80286 a 16 bit; clock 10/6 MHz; memoria RAM da 1Mb. Un hard disk da 40Mb; un floppy disk drive da 1,2Mb. Sistema operativo MS-DOS3.2, GW-Basic. Scheda video monocromatico/colore EGA WONDER di serie. Monitor monocromatico a fosfori verdi da 14". Porta seriale RS232 e parallela Centronics; tastiera avanzata 102 tasti con 12 tasti funzione	**L. 4.490.000**
PC 60/40	Microprocessore 80386 a 32 bit; clock 8/16 MHz; memoria RAM da 2,5 MB; hard disk da 40 Mb; 1 floppy disk drive da 1,2 Mb; sistema operativo MS-DOS 3.20; scheda EGA WONDER di serie; monitor ADI monocromatico a fosfori verdi da 14"; 2 porte seriali RS232C e 2 parallele Centronics. In opzione: floppy disk drive da 3" 1/2 e 1.44 Mb. Tastiera avanzata 102 tasti con 12 tasti funzione	**L. 8.490.000**
PC 60/80	Stessa configurazione del PC 60/40 ma con hard disk da 80 Mb e, in più, MS-DOS WINDOWS 386 e mouse	**L. 9.990.000**

Amiga 2000	Microprocessore Motorola 68000; clock 7,16 MHz; 1MB RAM; 256KB ROM; kickstart in ROM. Uscita PAL; 4 uscite a 4 voci su 2 canali (stereofonia). Amiga DOS; Amiga multitasking. 7 slots di sistema (2 combinati Amiga - Ibm PC AT); 1 slot video; 1 accesso diretto CPU 86 pin. Compatibilità MS-DOS interna con schede Janus (XT/AT CARD). Possibilità di un secondo drive interno da 3" 1/2 e di un drive interno da 5" 1/4. Monitor a colori 1084 tastiera; mouse. Workbench; Extras. MONITOR ESCLUSO	**L. 2.005.000**
	Monitor	
2080	Monitor a colori ad alta risoluzione e lunga persistenza 14" con audio; 640x400 pixel; 4096 colori per A500/A1000/A2000/PC AT	**L. 630.000**
1084	Monitor a colori ad alta risoluzione 14" con audio; antiriflesso; 640x400 pixel; 4096 colori per A500/A1000/A2000/PC AT	**L. 545.000**
1402	Monitor monocromatico da 12"	**L. 255.000**
1450	Monitor monocromatico BISYNC	**L. • □**
1950	Monitor colore BISYNC	**L. • □**
A 2024	Monitor monocromatico alta risoluzione a pagina bianca	**L. • □**
	Stampanti	
MPS 1200P	Stampante a 9 aghi 120 cps bidirezionale 80 colonne. Interfaccia Commodore per PC 10/III PC 40 PC/60/80	**L. 480.000**
MPS 1250	Stesse caratteristiche del modello MPS 1200P, interfacce seriale RS232C e parallela Centronics di serie	**L. 495.000**
MPS 1500	Stampante a colori a 9 aghi 130 cps; bidirezionale; 80 colonne; carta in modulo singolo e trascinamento modulo continuo; 4 colori; per Amiga e PC	
	Unità dischi	
K 910	Floppy disk drive aggiuntivo da 3" 1/2 e 1.44 Mb per PC 10-III e PC 20-III	**L. 295.000**
A 1010	Floppy disk drive esterno da 3" 1/2	**L. 395.000**
A 2010	Floppy disk drive interno da 3" 1/2	**L. 290.000**
A 2092 + PC 5060	Hard disk da 20 Mb settorizzabile PC/Amiga + scheda controller per hard disk per A 2000	**L. 845.000**
A 2080 + A 2020	Scheda Janus XT compatibile + drive	**L. 895.000**
A 2286 + A 2020	Scheda Janus II AT compatibile + drive	**L. 1.285.000**
A 2620	Scheda 68020 con processore a 32 bit; 14,28 MHZ	**L. •**
A 2090 + A 2092	Hard disk 20 Mb; interfaccia SCSI per A 2000	**L. 1.025.000**
A 2090 + A 2094	Hard disk 40 Mb; interfaccia SCSI per A 2000	**L. 1.575.000**
A 590	Hard disk da 20 Mb + fast RAM per A 500	**L. • □**
	Accessori	
PC EXP1	Box esterno per espansione PCI; può contenere 3 schede aggiuntive e hard disk alimentazione e cavi compresi	**L. 590.000**
A 2052	Scheda di espansione RAM da 2 Mb per A 2000 (max 4)	**L. 590.000**
A 2060	Scheda video/RF molulatore per TV e VCR per A2000	**L. 138.000**
A 2058	Espansione di memoria da 8 Mb RAM per A2000	**L. 3.760.000**
A UNIX	Scheda UNIX utilizzabile con A2620 (68020 board)	**L. • □**
1352	Mouse due tasti collegabile alla serie dei PC include Microsoft Mouse Driver	**L. 72.000**
A 2300	Genlock card semiprofessionale per A2000; permette di miscelare grafici con videocamere VCR; selettore per video Amiga, monitors esterni, proiettori; include software per titolazione	**L. 345.000 Δ**
A 2350	Professional Video Adapter Card per A 2000. Genlock professionale con freeze frame e digitalizzatore video incluse software di controllo per la gestione interattiva	**L. 1.645.000 Δ**

• prezzo da definire
Δ disponibile da giugno '88
□ disponibile da settembre '88

I primi 100 negozi selezionati dalla "Commodore" per la vendita dei suoi prodotti

Negozio	Indirizzo	Città	Telefono
Al Risparmio	*V.le Monza 204*	20128 Milano	Tel. 02-2573440
Braha Alberto S.D.F.	*Via Pier Capponi 5*	20145 Milano	Tel. 02-437468
E.D.S.	*C.so Porta Ticinese 4*	20123 Milano	Tel. 02-8322045
ESC	*Via Roggia Scagna 7*	20127 Milano	Tel. 02-2871300
Faref	*Via A. Volta 21*	20121 Milano	Tel. 02-650042
Giglioni	*Via D'Ovidio 8*	20131 Milano	Tel. 02-2360397
Giglioni	*V.le Luigi Sturzo 45*	20154 Milano	Tel. 02-654906
Logitek	*Via Golgi 60*	20133 Milano	Tel. 02-538931
Marcucci	*Via F.lli Bronzetti 37*	20129 Milano	Tel. 02-7386051
Melchioni	*Via P. Colletta 37*	20135 Milano	Tel. 02-57941
Messaggerie Musicali	*Galleria Del Corso*	20100 Milano	Tel. 02-50841
Newel	*Via Mac Mahon 75*	20155 Milano	Tel. 02-323492
Rivola	*Via Vitruvio 43*	20124 Milano	Tel. 02-6694160
F.lli Galimberti	*Via Naz. dei Giovi 28/36*	20030 Barlassina (Mi)	Tel. 0362-560625
P. Giorgio Ostellari	*Via Molino Arese 65*	20031 Cesano Mad.(Mi)	Tel. 0362-504392
P. Giorgio Ostellari	*Via Milano 300*	20233 Desio (Mi)	Tel. 0362-621042
GBC Italiana	*V.le Matteotti 66*	20092 Cinisello B. (Mi)	Tel. 02-6181801/ 6189391
	Via Petrella 6	20100 Milano	Tel. 02-203607/8
	Via Cantoni 7	20100 Milano	Tel. 02-437478
Casa Della Musica	*Via Indipendenza 21*	20093 Cologno Mon.(Mi)	Tel. 02-2542117
Penati	*Via Verdi 28/30*	20011 Corbetta (Mi)	Tel. 02-9779401
EPM System	*V.le Italia 12*	20094 Corsico (Mi)	Tel. 02-4407979
Centro Comp. Pandolfi	*Via Corridoni 18*	20025 Legnano (Mi)	Tel. 0331-546426
Computeam	*Via Vecellio 41*	20035 Lissone (Mi)	Tel. 039-481010
Futura	*Via Solferino 31*	20075 Lodi (Mi)	Tel. 0371-54457
MBM Inf. Sys.	*Corso Roma 112*	20075 Lodi (Mi)	Tel. 0371-53610
L'Amico del computer	*V.le Lombardia 17*	20077 Melegnano (Mi)	Tel. 02-9838341
Bit 84	*Via Italia 4*	20052 Monza (Mi)	Tel. 039-320813
I.C.O.	*Via dei Tigli 14*	20090 Opera (Mi)	Tel. 02-5242146
Comif	*Via Autolinee 10*	24100 Bergamo	Tel. 035-218553
Cordani	*Via dei Caniana 8*	24100 Bergamo	Tel. 035-258184
D.R.B.	*Via Borgo Palazzo 65*	24100 Bergamo	Tel. 035-237292
New Systems	*Via Paglia 36*	24100 Bergamo	Tel. 035-248109
Computer Team hi-tec	*Via Verdi 1/B*	24030 Carvico (Bg)	Tel. 035-790244
Ott. Optometr. Rovetta	*P.zza Garibaldi 6*	24065 Lovere (Bg)	Tel. 035-960705
A.I.S. International	*Via San Carlo 25*	24016 S.Pellegrino T.(Bg)	Tel. 0345-22662
Sisthema	*Via Roma 45*	24067 Sarnico (Bg)	Tel. 035-910750
Computer Center	*Via Cipro 62*	25125 Brescia	Tel. 030-223230
Informatica 2000	*Via Stazione 16/B*	25100 Brescia	Tel. 030-54015
Vigasio Mario	*Portici Zanardelli 3*	25100 Brescia	Tel. 030-59330/ 295858
Mister Bit	*Via Mazzini 70*	25043 Breno (Bs)	Tel. 0364-22835
Cavalli Pietro	*Via 10 Giornate 14B*	25030 Castrezzato (Bs)	Tel. 030-714013
Vietti Giuseppe	*Via Milano 1/B*	25032 Chiari (BS)	Tel. 030-7100206
Megabyte	*P.zza Duomo 17*	25015 Desenzano G.(Bs)	Tel. 030-9144880
Ditta Resi Rino	*Via XX Settembre 7*	25016 Ghedi (BS)	Tel. 030-901224
Info Cam	*Provinciale 38*	25050 Gratacasolo (Bs)	Tel. 0364-89379
Il Computer	*Via Indipendenza 90*	22100 Como	Tel. 031-240959
2M Elettronica	*Via Sacco 3*	22100 Como	Tel. 031-278227
Eletrongros	*Via L. Da Vinci 54*	22062 Barzanò (Co)	Tel. 039-957318
Ega	*Via Mazzini 42*	22065 Cassago Br.(Co)	Tel. 039-956307
Ega	*Via Aldo Moro 17*	22043 Galbiate (Co)	Tel. 0341-522028
Data Found Comp. Sh.	*Via A. Volta 4*	22036 Erba (Co)	Tel. 031-645761
Righi Elettronica	*Via G. Leopardi 26*	22077 Olgiate Com. (Co)	Tel. 031-946766
Fumagalli	*Via Cairoli 48*	22053 Lecco (Co)	Tel. 0341-863341
Cima Eelettronica	*Via Leonardo Da Vinci 7*	22053 Lecco (Co)	Tel. 0341-371106
Mondo Computer	*Via Giuseppina 11/B*	26100 Cremona	Tel. 0372-882079
Prisma	*Via Buoso Da Dovara 8*	26100 Cremona	Tel. 0372-437900
Telco	*P.zza Marconi 2/A*	26100 Cremona	Tel. 0372-31544
Elcom/GBC	*Via IV Novembre 56/58*	26013 Crema (Cr)	Tel. 0373-83393
Euroelettronica	*Via XX Settembre*	26013 Crema (Cr)	Tel. 0373-86966
Kofbaker	*Via Marchi 65/B*	26039 Vescovato (CR)	Tel. 0372-818601
Computer	*Galleria Fermi 7*	46100 Mantova	Tel. 0376-325616
32 Bit (Comp. St.)	*Via Cesare Battisti 14*	46100 Mantova	Tel. 0376-326770
Elettronica di Basso	*V.le Risorgimento 69*	46100 Mantova	Tel. 0376-329311
Poliware	*C.so C. Alberto 76*	27100 Pavia	Tel. 0382-22636
Log. Infor. Com. Shop	*V.le Monte Grappa 32*	27029 Vigevano (Pv)	Tel. 0381-81883
M. Visentin	*C.so Vitt. Emanuele 76*	27029 Vigevano (Pv)	Tel. 0381-83833
Computer Line	*Via G. Carducci 4*	29100 Piacenza	Tel. 0523-30691
Delta Computer	*Via Mart. della Resistenza 15/4*	29100 Piacenza	Tel. 0523-753318
Sover	*Via IV Novembre 60*	29100 Piacenza	Tel. 0523-34388
Cipolla Mauro	*Via Tremogge 25*	23100 Sondrio	Tel. 0342-213569
Fotonova		23010 S. Pietro Ber. (So)	Tel. 0342-492319
Dimeco Sistemi	*Via Garibaldi*	21100 Varese	Tel. 0332-237201
Il Centro Elettronico	*Via Morazzone 2*	21100 Varese	Tel. 0332-231006
Supergames	*Via Carrobbio 13*	21100 Varese	Tel. 0332-241092
Busto Bit	*Via Gavinana 17*	21052 Busto Arsizio (Va)	Tel. 0331-625034
Crespi Giuseppe & C.	*V.le Lombardia 59*	21053 Castellanza (Va)	Tel. 0331-503023
Computer Shop	*Via A. Da Brescia 2*	21013 Gallarate (Va)	Tel. 0331-798612
Lima Import-Export	*Via Clerici 196*	21040 Gerenzano (Va)	Tel.
J.A.C. nuove tecn.	*Via Matteotti 38*	21018 Sesto Cal. (Va)	Tel. 0331-923134
Bit Micro	*Via Mazzini 102*	15100 Alessandria	Tel. 0131-443252
West Records	*C.so Roma 85*	15100 Alessandria	Tel. 0131-441090
S.G.E. Elettronica	*Via Bandello 19*	15057 Tortona (Al)	Tel. 0131-867709
Record di Forina G.	*C.so Alfieri 166/3*	14100 Asti	Tel. 0141-34240
Il Diagramma Italia	*P.le Libertà 4*	12100 Cuneo	Tel. 0171-57067
Rossi Computers	*C.so Nizza 42*	12100 Cuneo	Tel. 0171-2339
Punto Bit	*C.so Langhe 26/C*	12051 Alba (Cn)	Tel. 0173-49833
SDI	*Via Vitt. Emanuele 250*	12042 BRA (CN)	Tel. 0172-421392
Aschieri Gianfranco	*C.so Eman. Filiberto 6*	12045 Fossano (CN)	Tel. 0172-62995
Curetti Augusto	*C.so Italia 3*	12054 Mondovì (Cn)	Tel. 0174-42014
Ditta Elettrogamma	*C.so Risorgimento 20*	28100 Novara	Tel. 0321-176358
Elcom	*C.so Mazzini 11*	28100 Novara	Tel. 0321-391293
Programma 3	*V.le Buonarroti 8*	28100 Novara	Tel. 0321-36367/ 399903
Punto Video	*C.so Risorgimento 391*	28100 Novara	Tel. 0321-477367
Computer	*Via Monte Zeda 4*	28041 Arona (No)	Tel.
All Computer	*C.so Garibaldi 106*	28021 Borgomanero (No)	Tel. 0322-844142
Micrologic	*Via Giovanni XIII 2*	28037 Domodossola (No)	Tel.
Elliott Computer Shop	*Via Don Minzoni 32*	28044 Intra (No)	Tel. 0323-43517
ABA Elettronica	*Via C. Fossati 5/P*	10100 Torino	Tel. 011-302065
Alex Computer e giochi	*C.so Francia 333/4*	10142 Torino	Tel. 011-7730184
C.D.M. Elettronica	*Via Marochetti 17*	10126 Torino	Tel. 011-636345/ 634900
Computing New	*Via Marco Polo 40/E*	10129 Torino	Tel. 011-501512
De Bug	*C.so Vitt. Emanuele II 22*	10100 Torino	Tel. 011-832986
Desme Universal	*Via San Secondo 95*	10100 Torino	Tel. 011-592551
F.D.S.	*Via Borgaro 86/D*	10100 Torino	Tel. 011-2168900
Computer Home	*Via San Donato 46/D*	10100 Torino	Tel. 011-4731196
Informatica Italia	*C.so Re Umberto 129*	10128 Torino	Tel. 011-501647
MT Informatica	*C.so Giulio Cesare 58*	10100 Torino	Tel. 011-850955/ 238803
New Bus. Computer	*Via Nizza 45*	10100 Torino	Tel. 011-214235
Radio Tv Mirafiori	*C.so Un. Sovietica 381*	10135 Torino	Tel. 011-616190/ 6197189
SMT Elettronica	*Via Bibiana 83/B*	10147 Torino	Tel. 011-218243
Paul e Chico Videos.	*Via Vitt. Emanuele 52*	10023 Chieri (To)	Tel. 011-9470295
Bit Informatica	*Via V. Emanuele 154*	10073 Cirié (To)	Tel. 011-9205455
HI-FI Club	*C.so Francia 92C*	10093 Collegno (To)	Tel. 011-4110256
ICS	*Stradale Torino 73*	10015 Ivrea (To)	Tel. 0125-251797
Cerrutti Mauro	*C.so Torino 234*	10064 Pinerolo (To)	Tel. 0121-70316
Eurex	*C.so Indipendenza 5*	10086 Rivarolo C.se (To)	Tel. 0124-27984
Ditta Elettrogamma	*C.so Bormida*	13100 Vercelli	Tel. 0161-53689
Elettronica	*Strada Torino 15*	13100 Vercelli	Tel. 0161-393163
C.S.I. Teorema	*Via Losana 9*	13051 Biella (Vc)	Tel. 015-28622
Fotostudio Trevisan	*Via XXV Aprile 24/B*	13014 Cossato (VC)	Tel. 015-921431
Studio Fotogr. Imarisio	*P.zza Martiri Libertà 7*	13039 Trino (Vc)	Tel. 0161-82081
F.lli Gatti	*Via Festaz 75*	11100 Aosta	Tel. 0165-35659

PRODOTTI SYSTEMS EDITORIALE

Software su cassetta

La voce III	L.12000
Raffaello	L.10000
Oroscopo	L.12000
Computer-Music	L.12000
Gestione familiare	L.12000
Banca dati	L.12000
Dichiarazione dei redditi (740/S)	L.16000
Matematica finanziaria	L.20000
Analisi di bilancio	L.20000
Arredare (richiede linguaggio Simon's Basic)	L.10000

Software su disco

Ms-Dos & Gw-Basic	L.25000
Ms-Dos & Gw-Basic (con prova di acquisto cassetta)	L.15000
La voce III	L.12000
Raffaello	L.10000
Oroscopo	L.12000
Computer-Music	L.12000
Gestione familiare	L.12000
Banca dati	L.12000
Dichiarazione dei redditi (740/S)	L.24000
Matematica finanziaria	L.20000
Analisi di bilancio	L.20000
Arredare (richiede linguaggio Simon's Basic)	L.20000
Graphic Expander C/128 in modo 80 colonne	L.27000
Linguaggio macchina + Routine grafiche	L.12000

Offerta "Commodore speciale L.M." + dischetto L.16000

Libri

64 programmi per il Commodore 64	L. 4800
I miei amici C/16 e Plus/4	L. 7000
Strategie vincenti per Commodore 64	L. 5800
62 programmi per Vic 20, C/16 e Plus/4	L. 6500
Utilities e giochi didattici	L. 6500
Tutti i segreti dello Spectrum	L. 7000
Simulazioni e test per la didattica	L. 7000
Impara giocando il Basic dello Spectrum	L. 7000
Micro Pascal per Commodore 64/128	L. 7000
Dal registratore al drive del C/64	L. 7000
Ada	L. 5000
Il linguaggio Pascal	L. 5000

Directory

Ciascun dischetto	L.12000

Arretrati

Ciascun numero arretrato di Commodore Computer Club	L. 5000
Ciascun numero arretrato di Personal Computer	L. 5000
Ciascun numero arretrato di VR Videoregistrare	L. 5000

Per un ottimale utilizzo del software "Matematica finanziaria" è opportuna la lettura degli articoli relativi pubblicati sui N.13, 14, 15 della rivista "Commodore" e sui N.1, 2 e 3 della Rivista Personal Computer.

Per un ottimale utilizzo del software "Analisi di Bilancio" è opportuna la lettura degli articoli relativi pubblicati sui N.2, 3, 5 della Rivista Personal Computer.

Per un ottimale utilizzo del software "Linguaggio Macchina e Routine grafiche per C/64" è opportuna la lettura del fascicolo "Commodore Speciale" appositamente dedicato.

Coloro che desiderano procurarsi i prodotti della Systems Editoriale devono inviare, oltre alla cifra risultante dalla somma dei singoli prodotti, la cifra di L.3000 per spese di imballo e spedizione, oppure L.6000 se si preferisce la spedizione per mezzo raccomandata.

Sconti e agevolazioni

Le spese di imballo e spedizione sono a carico della Systems se ciascun ordine è pari ad almeno L.50000 (di listino).

Gli abbonati hanno diritto allo sconto del 10% e alla spedizione gratuita se la somma totale raggiunge la cifra di L.50000 (di listino).

Oltre alla spedizione gratuita, viene praticato uno sconto del 10% (per gli abbonati è del 20%) se la cifra raggiunta per ciascun ordine raggiunge le L.100000 (di listino).

Abbonamenti

Commodore Computer Club (11 fascicoli)	L.45000
Personal Computer (11 fascicoli)	L.40000
Commodore Computer Club + Personal Computer (11 + 11 fascicoli)	L.70000
VR Videoregistrare (12 numeri)	L.45000

N.B.: la cifra per gli abbonamenti non può essere conteggiata per ottenere gli sconti e le agevolazioni di cui sopra.

Non è assolutamente possibile inviare materiale contrassegno.

Compilate un normale modulo di C/C postale indirizzando a:

C/C postale N. 37952207
Systems Editoriale
Viale Famagosta, 75
20142 Milano

Non dimenticate di indicare chiaramente, sul retro del modulo (nello spazio indicato con "Causale del versamento") non solo il vostro nominativo completo di recapito telefonico, ma anche il materiale desiderato.

In ogni caso sarebbe opportuno inviare la presente scheda, debitamente compilata, allegando la fotocopia della ricevuta del versamento effettuato.

Chi volesse ricevere più celermente la confezione deve inviare la somma richiesta mediante assegno circolare oppure normale assegno bancario (non trasferibile o barrato due volte) intestato a: Systems Editoriale - Milano.